Junior

Общие вопросы

1. В чем заключаются основные принципы ООП?
2. Что такое сложность алгоритма?
3. Код работает неправильно. Что делать?
4. Объясните такие структуры данных, как стек и очередь.
5. Какие книги, связанные с программированием, вы читали? Что почерпнули для себя?
6. Что интересного нашли в новых стандартах C++ 17 C++ 20?
7. Что такое таблица ASCII?
8. Что такое Unicode?
9. Что такое паттерны проектирования и для чего их используют?
10. Паттерны Singleton, Strategy, Template-Method, Decorator?
11. Для чего нужны модульные тесты?
12. Как разница между модульными и интеграционными тестами?
13. Что такое TDD?

Метапрограммирование

14. Что такое шаблонный класс и шаблонная функция?
15. Что такое конструкторы? Какие типы вы знаете?
16. Может ли конструктор быть шаблонной функцией?
17. Может ли виртуальная функция быть шаблонной?
18. Что такое инстанциация шаблона?
19. Что такое специализация шаблона? Частичная специализация шаблона?
20. Расскажите об имплементации шаблонных классов в cpp-файле.

Препроцессор и компиляция

21. Как проходит процесс компиляции cpp-файлов в бинарный файл?
22. Что такое препроцессор?
23. Как работает препроцессор?
24. Какие знаете его команды?
25. Как работает директива include?
26. Как работает директива define?
27. Что именно линкует линкер?
28. Что такое оптимизация компилятора?
29. Что такое флажки компиляции?
30. Как защитить хедер от повторного включения?
31. Что делает директива include?
32. Как работают макросы?

C

33. Как static влияет на глобальные / локальные переменные?
34. Как const влияет на переменную?

35. Какие варианты использования extern вы знаете?
36. Какие варианты использования volatile вы знаете?
37. Какие есть битовые операции?
38. Что такое булева алгебра?
39. Расскажите об этапах разработки библиотеки или программы.
40. Что такое алгоритмы сортировки и какие вы знаете?
41. Какие алгоритмы работы со строками знаете?
42. Какие алгоритмы на графах знаете?
43. Где может храниться переменная?
44. Какая разница между calloc и malloc?
45. Для чего используют realloc?
46. Что такое указатель?
47. Какой размер имеет указатель и от чего он зависит?
48. Какие есть операции с указателями?
49. Что такое struct?
50. Как определить размер структур?
51. Что такое выравнивание в структурах?
52. Что такое union?
53. Каков размер union?

C++ / OOP

54. Что такое класс?
55. Какие основные типы данных в C++?
56. Что такое инкапсуляция? Как она реализуется в C++?
57. Какие есть встроенные типы в C++?
58. Что такое enum?
59. Как соотносится класс и объект?
60. Какая разница между структурой и классом?
61. В чем разница между private / protected / public и где они используются?
62. Какие методы класса являются стандартными для класса?
63. Что такое абстрактный класс и зачем он нужен?
64. Сколько занимает памяти объект пустой класс class A{}; ?
65. Что станет с функцией, если к ней добавить ключевое слово static? В контексте члена класса? В контексте метода класса?
66. Какие особенности статических полей класса?
67. В чем особенность константных методов-членов класса?
68. Как изменить поле класса в константном методе класса?
69. Какие методы можно вызвать из константных объектов?
70. Что такое куча и стек? Различия, принцип работы.
71. В чем разница между указателем и ссылкой?
72. Для чего нужен указатель на функцию? Как его объявить?
73. Что будет, если забыть вызвать delete? Когда освободится та память?
74. Что такое умный указатель? Какие умные указатели есть в стандартной библиотеке?
75. Как работает std::unique_ptr?
76. Как работает std::shared_ptr?

77. Расскажите о константности переменной, ссылки, указателя. Что такое константный указатель и указатель на константу? Размер указателя в памяти?
78. Расскажите о передаче аргументов по значению, ссылке и указателю.
79. Расскажите о порядке вычисления аргументов функции.
80. Что случится, если вернуть ссылку на временный объект?
81. Что такое перегрузка функции? Виды перегрузки.
82. Что такое явное и неявное приведение типов в C++? Расскажите о функциях явного приведения типов в C++.
83. Что такое инициализация переменной в if?
84. Что такое ленивые вычисления в C++?
85. Расскажите о циклах for и range-for.
86. Что делает ключевое слово auto? auto-определение return-типа, аргументов функции?
87. Чем отличаются delete и delete[]? Что случится, если вызвать delete у объекта, созданного через new[]?
88. Обработка ошибок в C++. Какие конструкции используют при обработке exception?
89. Можно ли выбрасывать exception из конструктора? Какие поля будут сконструированы, какие поля будут разрушены?
90. Что такое memory leak?
91. Можно ли выбрасывать exception с деструктора?
92. Как отловить деление на 0 в C++?
93. Как работают константные методы?
94. Что такое лямбда-функция в C++? Как получить доступ к переменным во внешней области видимости?
95. Для чего использовать namespace, anonymous namespace?
96. Как вызвать объект с nested namespace?
97. Как работают inline-функции? Может ли такая функция быть рекурсивной?
98. Что такое полиморфизм?
99. Для чего используется наследование?
100. Какие бывают типы наследования?
101. Для чего используют виртуальное наследование?
102. Как можно решить проблему ромбовидного наследования без использования виртуального наследования?
103. Что случится, если класс-наследник передать по значению в функцию, которая принимает базовый класс?
104. Что случится, если унаследоваться от базового класса, не имеющего виртуального конструктора?
105. Что случится, если вызвать переопределенную virtual function из конструктора? Может конструктор быть виртуальным?
106. Может ли иметь имплементацию pure virtual function? Что случится, если вызвать pure virtual function из конструктора?
107. Какие методы генерируются для класса по умолчанию? В каком случае такие методы не будут генерироваться? Как заставить компилятор добавить / удалить эти методы?
108. Как запретить наследовать класс?

109. Каков порядок конструирования и разрушения классов в иерархии? Порядок инициализации полей класса?
110. Какие есть способы инициализации полей класса?
111. Может ли деструктор быть виртуальным?
112. Что делает ключевое слово virtual?
113. Для чего используют виртуальный деструктор?
114. Что такое глубокое копирование?
115. Что такое виртуальные функции и зачем они нужны?
116. Как защитить объект от копирования?
117. Что такое семантика перемещения?

STL / Algorithms

118. Из чего состоит STL?
119. Какие алгоритмы применяли с STL? В чем преимущество использования алгоритмов перед собственноручно написанными функциями?
120. Расскажите о контейнерах стандартной библиотеки vector, list, map, unordered_map.
121. Какие знаете типы итераторов? Чем они отличаются? В каких контейнерах используются?
122. Какая разница между std::set, std::map std::unordered_multimap?
123. Что такое идиома remove-erase?
124. Как получить наименьшее значение типа?
125. Какая разница между std::map и std::hashmap?
126. Как подсчитать количество элементов в std::list?
127. Что такое сложность алгоритма и от чего она зависит?
128. В чем разница между vector и list и в каких случаях их лучше использовать?

Многопоточность

129. Что вам известно о многопоточности?
130. Что общего и отличного в процессах и потоках?
131. Как синхронизировать передачи информации между потоками?
132. Какая разница между мьютекс и семафором?
133. Что такое deadlock?
134. Является ли C++ thread-safe?
135. Что такое race-condition?
136. Как избежать состояния гонки?
137. Что такое атомарная операция?
138. Как работать с std::mutex?

Networking

139. Что такое сокет?
140. Какие операции можно делать с сокетом?
141. Какая информация нужна, чтобы создать сокет?
142. Какие бывают модели сетей?

143. Расскажите об уровнях модели OSI.
144. Расскажите об уровнях модели TCP/IP.
145. Что такое IP-адрес?
146. Для чего используется маска подсети?
147. Какая разница между IPv4 и IPv6?
148. Сколько памяти необходимо для хранения IPv4?
149. Зачем порт?
150. Сколько максимально может быть портов?
151. Какая разница между TCP и UDP?
152. Для чего такой ненадежный UDP-протокол?

OS / Linux

153. Что такое менеджер пакетов?
154. Какие бывают менеджеры пакетов?
155. Какие бывают дистрибутивы Linux?
156. Что такое PID?
157. Для чего используют файловые дескрипторы?
158. Расскажите о стандартных файловых дескрипторах процесса.
159. Что такое Pipe?
160. Что такое Named Pipe?
161. Что такое UID?
162. Расскажите о командах bash.

SCM / CI / CD

163. Какие есть виды SCM?
164. Для чего используют системы контроля версий?
165. Какие есть команды git?
166. Какие этапы во время коммита изменений?
167. Разница между git fetch и git pull?
168. Какие есть этапы решения merge conflict?

Практические задания

169. Посчитайте количество единиц в произвольном числе.
170. Структура по типу «односвязный список». Напишите функцию, которая разворачивает список. То есть первый элемент становится последним, а последний — первым.
171. Напишите реализацию функции int atoi(const char * str); преобразования строки в число.
172. Для структуры типа односвязный список напишите функцию вставки элемента.
173. Реализуйте класс vector.
174. Реализуйте бинарный поиск в массиве.
175. Реализуйте любую сортировку.
176. Реализуйте макрос для сравнения двух строк.
177. Реализуйте реверс строк.
178. Реализуйте перевода числа из строки в int.

179. Реализуйте подсчет слов в предложении.
180. Реализуйте подсчет чисел Фибоначчи.
181. Найдите такие элементы двух массивов, которые встречаются только в каждом из них. Желательно использовать STL.
182. Удалите из unordered_map элементы, которые делятся на 2, и выведите ключи этих элементов.
183. Напишите класс для логирования, который мог бы логировать в консоль или файл.
184. Напишите функцию для определения, является ли указанный год високосным.
185. Напишите функцию для определения, является ли заданное слово палиндромом.
186. Напишите реализацию паттерна Singleton.
187. Напишите реализацию std::vector с операциями: push_back, push_front, pop_back, pop_front, size, clear.
188. Напишите рекурсивный поиск значения в дереве бинарного поиска.
189. Напишите функцию, которая проверяет, является ли дерево сбалансированным.
190. Напишите функцию для поиска уникального элемента в массиве.

Middle

Общие вопросы

1. Какие курсы прошли или книги прочитали за этот год? Чему научились?
2. Что нравится и не нравится в C++? Чего не хватает?
3. Что интересного нашли в новых стандартах C++ 17 C++ 20 (конкретные фичи)?
4. Расскажите о фичах, которые появились в разных версиях языка.
5. Расскажите о модели памяти, которая появилась в стандарте C++ 11.
6. Что такое сериализация? Какие библиотеки знаете?
7. Какие знаете паттерны проектирования?
8. Что такое операционная система? Какие существуют типы по назначению?
9. Назовите основные составляющие и принципы работы ОС Linux в качестве примера системы общего назначения.
10.Что такое SFINAE и PIMPL?
11. Назовите порождающие, структурные и поведенческие паттерны программирования и приведите примеры их использования.

Препроцессор и компиляция

12. Расскажите о системах автоматизации билд-процесса.
13. В чем разница между статической и динамической библиотеками?
14. В чем разница между исполнительным файлом и динамичной библиотекой?

15. Что такое DLL hell?
16. Что такое флажки компиляции (fPIC)?
17. В чем разница между дебажной и релизной сборками?
18. Что нужно для использования сторонней библиотеки?
19. Что такое internal linkage?

C

20. Что будет, если дважды вызвать free?
21. Как происходит вызов функции?
22. Как происходит передача параметров в функцию?
23. Как обрабатывается константность переменных?
24. Что означает ключевое слово inline?
25. Для чего используют выравнивания, можно ли его контролировать?
26. Расскажите о битовых полях.
27. Для чего нужен extern «C»?
28. Что будет, если в двух файлах сделать функцию с одинаковым именем и параметрами? На каком этапе возникнет ошибка?
29. Как экспортировать / импортировать функции из динамической библиотеки?
30. Какая разница между C-style приведением типов и C++ приведением?

C++

31. Что такое явное и неявное приведение типов в C++? Зачем делать explicit-конструктор?
32. Что такое Uniform initialization? Aggregate initialization?
33. Что такое Reference to temporary object? Как продлить время жизни временного объекта?
34. Что такое делегирующий конструктор?
35. Что такое список инициализации?
36. Каков порядок инициализации полей класса? Что случится, если конструктор инициализирует поля в другом порядке?
37. Что случится, если инициализировать поле другим полем?
38. Что такое copy elision? Сколько раз будет вызван конструктор / деструктор у объекта, который возвращают по значению?
39. Что такое move-семантика?
40. В каких случаях будет сгенерирован конструктор копирования?
41. Чем отличается конструктор копирования от оператора присваивания?
42. При каких условиях в конструкторе можно выбросить exception?
43. Что такое конструктор по умолчанию? Для чего нужны default и delete?
44. Чем отличается интерфейс от абстрактного класса?
45. Какие виды полиморфизма в C++?
46. Как реализовано наследование в большинстве компиляторов?
47. Множественное наследование: за и против.
48. Виртуальное наследование и порядок конструирования.
49. Зачем использовать override?

50. Какие есть правила вывода типа при использовании auto? В каких случаях auto может привести к нежелательному копирования объекта?
51. Расскажите обо всех возможных способах использования ключевого слова static в C++. Что такое static initialization order fiasco?
52. Что делает вызов throw; в блоке catch?
53. Чем отличается constexpr от const?
54. Что такое const correctness?
55. В каком случае можно использовать const_cast?
56. Что такое ключевое слово mutable и когда его нужно использовать?
57. Что такое ключевое слово friend и когда его нужно использовать?
58. Расскажите о лямбда-выражениях в C++ и доступе к переменным во внешней области видимости, захвате this в лямбду и времени жизни лямбды и захваченных переменных.
59. Что такое функтор? Напишите пример.
60. Что такое специализация шаблона?
61. Что такое dynamic_cast и run-time type identification?
62. Что такое exception? Как бросить и поймать?
63. Что будет, если бросить exception из конструктора? А из деструктора?
64. Что будет, если не поймать exception?
65. Что произойдет, если exception выйдет за пределы блока noexcept функции?
66. Для чего можно использовать приватное наследование?
67. Что такое контракт функции?
68. Что такое vptr и vtable?
69. Где содержится vptr?
70. Где содержится vtable?
71. Какая разница между overload и override?
72. Как компилятор различает члены класса и обычные переменные в функциях?
73. Зачем используют exceptions?
74. Что такое блоки try-throw-catch?
75. Расскажите о логике catch-блоков.
76. Что такое move constructor?
77. В чем разница между константным методом и неконстантным?
78. Что такое O-нотация и как определить сложность любого алгоритма?
79. Что такое таблица виртуальных методов?
80. Какие функции класса автоматически генерирует компилятор, если их определить?
81. Что такое выравнивание данных?
82. Что такое exception?
83. Какие есть стандартные контейнеры и на основе каких структур они построены?
84. Что такое Undefined behavior? Приведите примеры.
85. Как определить, что в программе есть memory leak?
86. Для чего нужен std::make_shared? Чем он лучше создания std::shared_ptr через конструктор?
87. Что будет, если выделить один объем памяти, а записать больше?
88. Что такое переполнение stack?

Паттерны проектирования

89. Зачем нужны паттерны? Какие типы паттернов различают?
90. Недостатки паттерна Singleton. Когда он уместен?
91. Преимущества и недостатки PIMPL.
92. В чем разница между паттерн-фабрикой и фабричным методом? Когда использовать какой из них?
93. Что такое паттерн Observer?
94. Как контролировать состояние программы? Машину состояний? Паттерн состояние?
95. Что такое паттерн Visitor?

Метапрограммирование

96. Какие есть правила вывода типа в шаблоне?
97. Чем отличается using от typedef?
98. Сколько памяти занимает произвольная структура? Что такое выравнивание объекта?
99. Почему пустая структура занимает 1 байт? Какая минимальная единица адресации в C++?

OOP / OOD

100. Что такое SOLID? Что означает каждый из этих принципов?
101. Расскажите о паттернах проектирования.
102. Что такое Dependency Injection? Приведите пример.
103. Какие преимущества и недостатки функционального подхода?
104. Что такое принцип RAII?
105. Что такое принцип DRY?
106. Что такое принцип KISS?
107. Какие преимущества композиции перед наследованием?

STL / Algorithms

108. Какие алгоритмы с STL использовали? Каких не хватает?
109. Какими особенностями должен обладать класс, чтобы он был итератором?
110. Какие бывают итераторы?
111. Расскажите о инвалидации итераторов.
112. Как оптимизировать удаление элемента из середины вектора?
113. Как реализован vector?
114. Как реализован list?
115. Как расширить STL-контейнеры?
116. Какие есть алгоритмы в STL?
117. В чем разница между vector, deque, list, set e STL?
118. Когда надо использовать map? Когда — unordered_map? Какова сложность поиска и вставки в этих контейнерах?
119. Как проверить, есть ли в контейнере элементы? Почему вызов container.size() является плохой практикой?

120. Что такое exception safety guarantee? Какую exception safety guarantee имеют STL-контейнеры?
121. Расскажите о типах умных указателей и подсчете ссылок в них.

Многопоточность

122. Является ли C++ thread-safe?
123. В чем разница между многопоточностью и асинхронностью?
124. Что такое многопоточность? Какую функциональность предоставляет C++ для разработки многопоточных приложений? Каковы основные проблемы многопоточных приложений?
125. Как передать информацию между несколькими процессами?
126. Как синхронизировать между собой несколько процессов?
127. Какие есть особенности работы с shared memory?
128. Как работает spinlock?
129. Какие вы знаете особенности использования recursive mutex?
130. Расскажите о read-write mutex.
131. Что такое race-condition? Взаимная блокировка? Что такое критическая секция?
132. Какие избежать состояния гонки?
133. Чем отличается мьютекс от семафора?
134. Какие примитивы синхронизации реализованы в C++? Преимущества lock_guard?
135. Что случится, если exception выйдет за пределы потока? Какие инструменты есть для безопасной асинхронности в C++?
136. Чем отличается std::launch::async от std::launch::deferred?
137. Что такое атомарная операция? std::atomic?
138. Как работать с std::conditional_variable?
139. Как создать поток с помощью std::thread?
140. На сколько потоков лучше разбить задачу? От чего это зависит?
141. Как работать с std::async?
142. Thread-safe гарантии контейнеров в C++. В чем недостаток интерфейса front() + pop_fornt()?

Networking

143. Что такое TCP handshake?
144. В чем разница между TCP и UDP?
145. Расскажите о протоколах верхнего уровня.
146. В чем разница между HTTP и HTTPS?
147. Расскажите о SSL/TLS handshake.

SCM / CI / CD

148. Расскажите о процессах CI.
149. Как отредактировать коммит?
150. Расскажите об интерактивном rebase.
151. Какие могут быть способы дебаггинга кода?
152. Для чего нужны Unit test? Чем они отличаются от Functional Test?

153. Как тестировать код? Какой фреймворк используете?
154. Какие библиотеки знаете для написания тестов?
155. Что такое mock?
156. Сколько тестов нужно написать на одну функцию?
157. Что такое побочный эффект, идемпотентность и чистые функции?
158. Что такое контейнеризация и в чем ее преимущества и недостатки? Что такое Docker или иной инструмент контейнеризации?
159. Что такое CI/CD и какие преимущества дает разработчику?
160. Каковы принципы итеративных методологий?
161. Какие преимущества и недостатки code-convention?

Практические задания

162. Напишите максимально корректную реализацию класса string с конструктором копирования и оператором присваивания.
163. Напишите реализацию очереди.
164. Реализуйте функцию, которая за один проход найдет уникальный элемент в контейнере.
165. Напишите thread-safe пул потоков.
166. Напишите игру «Жизнь» в ООП-стиле.
167. Напишите класс, который получает из базы список товаров по фильтру и показывает на консоли. Напишите тесты для него.
168. Любая задача на написание кода, чтобы проверить умение проектировать интерфейсы и придерживаться принципов SOLID, DRY, KISS.
169. Напишите свою реализацию std::atomic.
170. Напишите программу для анализа графов: нахождение циклов, deadlock-состояний, циклов, недоступных состояний.
171. Напишите программу, проверяющую, что в системе запущен только один ее экземпляр. Решение должно быть cross-platform.
172. Проанализируйте C++ код с точки зрения качества: выявить потенциальные memory leak, нерациональное использование STL-контейнеров, алгоритмов, неоптимальные конструкции и тому подобное.
173. Напишите код для решения судоку.
174. Напишите код, который найдет зацикливание в односвязном списке.

Senior

Общие вопросы

1.Как вы понимаете SOLID?
2. Как разработать систему плагинов на C++?
3. Что такое RPC? Какие библиотеки знаете?
4. На что обращать внимание при проведении code review?
5. Какие есть проблемы при написании кроссплатформенного кода? На что обращать внимание?
6. Что делать, если код работает медленно?

7. Какие есть способы и методология измерения быстродействия кода? Как можно устранить / уменьшить влияние замеров на быстродействие?
8.Что такое SFINAE? Для чего используется?
9. Что такое метапрограммирование? С помощью чего реализуется на C++?
10. Как использовать variadic templates?
11. Как тестировать закрытые методы?
12. Как считать покрытие тестами? Нужно ли это делать?
13.Что такое cache miss и как это выявить?
14. Что такое SIMD-инструкции? Каковы необходимые условия и способы их использования?
15. Что такое покрытие кода и как оно обеспечивается?
16. Опишите принципы lock-free структур данных и свой опыт работы с ними.

Препроцессор и компиляция

17. Расскажите о построении билд-системы.
18. Как работать с билд-системами: Make, CMake.
19. Как интегрировать third-party в проект?
20. Что такое барьеры памяти?
21. Расскажите о работе с сырыми указателями и ручном управлении памятью.
22. Что такое статический анализатор кода? Какие знаете?
23. Что такое динамический анализатор кода? Какие знаете?
24. Проект медленно собирается. Как можно ускорить?

C / C++

25. Расскажите об использовании realloc в контейнерах.
26. Как работают шаблоны?
27. Расскажите о специализации шаблонов.
28. Как работает RTTI?
29. Можно ли использовать exception в конструкторе / деструкторе?
30. Что такое rvalue и lvalue?
31. В чем особенности контейнеров std::set, std::map, std::unordered_map, std::hash?
32. Что такое placement new? Для чего используют? Как сделать placement delete?
33. Как размещается в памяти класс с множественным наследованием и виртуальными функциями?
34. Как работают точки останова?
35. Что такое уязвимости? Механизм их работы?
36. Как написать собственный std::shared_ptr?
37. Что такое curiously recurring template pattern?
38. Опишите назначение и принцип работы std::shared_ptr, std::unique_ptr и std::weak_ptr.
39. Каково назначение и различия использования std::variant и std::any?
40. Какие улучшения получил std::search в C++ 17?

41. Что такое copy elision и когда становится возможным? Какие особенности для разных стандартов?
42. Что такое Return Value Optimization?

OOP / OOD

43. Поясните принципы SOLID.
44. Поясните принципы KISS.
45. Поясните принципы YAGNI.
46. Какие есть подходы к оптимизации кода?
47. На что стоит обращать внимание при code review?
48. Какие есть паттерны проектирования? Почему не советуют использовать Singleton?
49. Что такое статический полиморфизм?

STL / Algorithms

50. Когда std::vector может использовать std::move?
51. Расскажите о своем любимом алгоритме поиска.
52. Что такое lock-free и wait-free алгоритмы? В чем их отличия и способы реализации?
53. Опишите назначения execution policy для параллельных алгоритмов.

Многопоточность

54. Расскажите о построении API, рассчитанных на многопоточное использование.
55. В чем разница между kernel-level и user-level потоками?
56. Что такое coroutine?
57. Что делает спецификатор thread_local?
58. Как реализовать синхронизацию в задаче producer-consumer?
59. Как синхронизироваться между различными процессами?

SCM / CI / CD

60. Расскажите о настройке процесса менеджмента ветвей репозитория.
61. Расскажите о стратегии ветвления.

Практические задания

62. Напишите базовую реализацию std::shared_ptr.
63. Реализуйте алгоритм сортировки.
64. Реализуйте алгоритм хеширования.
65. Реализуйте shared_ptr с расширением для weak_ptr.
66. Реализуйте простейший producer-consumer, используя условные переменные.
67. Опишите как можно подробнее, что происходит в системе, когда приложение делает сетевой запрос.