Вопросы к дифференцированному зачету ООП зима 2025–2026

1. Умные указатели. unique_ptr — назначение, особенности использования, семантика владения, передача и возврат из функций, перемещение, ситуации применения.
2. Умные указатели. shared_ptr — подсчёт ссылок, особенности совместного владения, типичные ошибки использования.
3. Умные указатели. weak_ptr — назначение, предотвращение циклических зависимостей, работа с lock(), жизненный цикл наблюдаемого объекта.
4. Лямбда-функции — синтаксис, области применения, особенности хранения состояния, использование в алгоритмах STL. Списки захвата — виды захватов, ограничения, взаимодействие с внешними переменными.
5. Основы ООП — инкапсуляция, наследование, полиморфизм: назначение, преимущества, примеры реализации в C++.
6. Класс и структура — сходства и различия, область видимости по умолчанию, использование в проектировании данных.
7. Конструкторы и деструкторы — назначение, порядок вызова, автоматическое управление ресурсами. Правило трёх и правило пяти.
8. Статические поля и методы класса — назначение, особенности хранения, инициализация, области применения.
9. Наследование — виды наследования, порядок вызова конструкторов, особенности доступа, проблемы ромбовидного наследования.
10. Абстрактные классы — назначение, чисто виртуальные методы, интерфейсы, ограничения на создание объектов.
11. Дружественные функции и классы — назначение, механика доступа к закрытым членам, области применения и риски.
12. Полиморфизм — статический и динамический полиморфизм, виртуальные методы, механизм позднего связывания.
13. Виртуальная таблица (vtable) — принцип работы, структура, влияние на размер объекта, время вызова виртуальных функций.
14. Перегрузка операторов — возможности языка, операторы, которые можно/нельзя перегружать, требования к корректной семантике.
15. Перегрузка операторов присваивания, копирования и перемещения — особенности реализации, работа с ресурсами.
16. Функции-друзья и классы-друзья — назначение, преимущества и недостатки использования.
17. Пространства имён — назначение, предотвращение конфликтов имён, организация больших проектов.
18. SOLID: принцип единственной ответственности — назначение, влияние на архитектуру классов.
19. SOLID: принцип открытости–закрытости — расширяемость без модификации кода, средства реализации в C++.
20. SOLID: принцип подстановки Лисков — корректность наследования, проектирование базовых интерфейсов.
21. SOLID: разделение интерфейсов — минимальные интерфейсы, зависимости клиентов.
22. SOLID: инверсия зависимостей — абстракции вместо реализаций, применение в больших проектах.
23. Порождающие паттерны проектирования — назначение, примеры, сценарии использования.
24. Структурные паттерны проектирования — задачи, примеры, преимущества.
25. Поведенческие паттерны проектирования — особенности, области применения.