http://neerc.ifmo.ru/wiki/index.php?title=%D0%A1%D1%82%D0%B0%D1%80%D1%8B%D0%B5\_%D0%B2%D0%BE%D0%BF%D1%80%D0%BE%D1%81%D1%8B\_%D0%BD%D0%B0\_%D1%8D%D0%BA%D0%B7%D0%B0%D0%BC%D0%B5%D0%BD\_%D0%BF%D0%BE\_C%2B%2B

1. Что такое класс? Чем отличается класс от объекта?
2. Разрешается ли объявлять указатель на объект?
3. Допускается ли передавать объекты в качестве параметров, и какими способами? А возвращать как результат?
4. Чем класс отличается от структуры? Можно ли использовать ключевые слова public и private в структуре?
5. Объясните принцип инкапсуляции. Обязательно ли делать поля класса приватными?
6. Для чего используются ключевые слова public и private? Существуют ли ограничения на использование public и private в классе?
7. Что такое метод? Как вызывается метод? Что такое параметры значения по умолчанию?
8. Как определить метод непосредственно внутри класса и вне класса? Чем эти определения отличаются?
9. Что обозначается ключевым словом this?
10. Зачем нужны константные методы? Чем отличается определение константного метода от обычного?
11. Объясните принцип полиморфизма.
12. Можно ли перегружать операции для встроенных типов данных?
13. Можно ли определить новую операцию?
14. Перечислите особенности перегрузки операций через методы класса. Чем отличается перегрузка через функции от перегрузки через методы класса?
15. Какой результат должны возвращать операции с присваиванием?
16. Какие операции разрешается перегружать только как методы класса?
17. Дайте определение дружественной функции. Как объявляется дружественная функция? А как определяется?
18. Дайте определение конструктора. Каково назначение конструктора? Перечислите отличия конструктора от метода.
19. Сколько конструкторов может быть в классе? Допускается ли перегрузка конструкторов? Какие виды конструкторов создаются по умолчанию?
20. Может ли конструктор быть приватным? Какие последствия влечет за собой объявление конструктора приватным?
21. Приведите несколько случаев, когда конструктор вызывается неявно.
22. Какой вид конструктора фактически является конструктором преобразования типов?

Видимо, любой неявный констркутор от одного аргумента.

1. Для чего нужны функции преобразования? Как объявить такую функцию в классе?

Что это??

1. Как запретить неявное преобразование типа, выполняемое конструктором инициализации?

Для этого есть ключевое слово explicit

1. Какие проблемы могут возникнуть при определении функций преобразования?

 ????

1. Для чего служит ключевое слово explicit?

Чтобы не позволить вызвать конструктор от одного аргумента неявно.

1. Влияет ли наличие целочисленных констант-полей на размер класса?

Они ничем не отличаются от обычных не-const полей.

1. Разрешается ли объявлять массив в качестве поля класса. Как присвоить элементам массива начальные значения?

Разрешается, надо написать что-нибудь вроде int a[] = {1, 2, 3};

1. Сколько операндов имеет операция индексирования []? Какой вид результата должна возвращать эта операция?

Операции индексирования нужен один аргумент - индекс. Возвращать T или T& в зависимости от того, можно ли менять объект.

1. Для чего нужны статические поля в классе? Как они определяются?

Это какие-то параметры класса. Определяются с помощью ключевого слова static

1. Как объявить в классе и проинициализировать статический константный массив?

struct S{static const int a[];}; const int S::a[] = {1, 2, 3};

1. Что такое выравнивание и от чего оно зависит? Влияет ли выравнивание на размер класса?

Это способ расположения данных для ускорения доступа к ним. Да, размер может увеличиться.

1. Дайте определение контейнера.

Это объект, который хранит "коллекцию" других объектов. Контейнер сам заботится о выделении памяти и других низкуровневых штуках, а нам предоставляет доступ к объектам, которые в нем находятся.

1. Какие виды встроенных контейнеров в С++ вы знаете?

vector, deque, set, map, list

1. Какие виды доступа к элементам контейнера вам известны?

Для чтения и(или) записи в одном/двух направлениях. Также есть произвольный доступ.

1. Чем отличается прямой доступ от ассоциативного?

Прямой(direct access) - это вроеде через оператор[], а что такое ассоциативный??

1. Перечислите операции, которые обычно реализуются для последовательного доступа к элементам контейнера.

Видимо, инкремент итератора.

1. Дайте определение итератора.

Это объект - обертка над указателем на какой-либо элемент контейнера. В нем должны быть реализованы хотя бы операции инкремента и разыменования.

1. Можно ли реализовать последовательный доступ без итератора? В чем преимущества реализации последовательного доступа с помощью итератора?
2. Что играет роль итератора для массивов С++?

Хм, указатели, видимо.

1. Что такое деструктор? Может ли деструктор иметь параметры?

Деструктор - то что будет вызвано после того, как объект выходит из своей области видимости для того, чтобы корректно освободить то, что выделяли в динамической памяти.

1. Почему для классов-контейнеров деструктор надо писать явным образом?

Потому что там используется динамическая память.

1. Допускается ли перегрузка деструкторов?

Нет

1. Что такое «глубокое копирование» и когда в нем возникает необходимость?

Если мы копируем объект, в котором есть поле-указатель, надо скопировать объект, на который указывает тот указатель, а не просто указатель.

1. Какое копирование осуществляет стандартный конструктор копирования?

Тупо копируются все поля.

1. Чем отличается копирование от присваивания?

Копирование - инициализация объектом того же типа. Присваивание - замена уже существующего объекта.

1. Объясните, почему в операции присваивания требуется проверка присваивания самому себе?

Потому что если мы работаем с ссылками, можно попортить объект при очистке памяти:

struct A{int\* a; A(){a = new int(10);} A & operator=(const A &b){delete a; a = new int(\*b.a); return \*this}

В этом коде если присваиваем самому себе, неявно мы изменяем b после вызова delete a, и теряем значение \*a.

1. Можно ли в качестве операции индексирования использовать операцию вызова функции ()? В чем ее преимущества перед операцией []?

Можно впринципе. Плюс, видимо, в том что можно передавать сколько хочешь аргументов.

1. Почему необходимо писать два определения операции индексирования? Чем они отличаются?

Первое определение возвращает ссылку на элемент, которых храним в контейнере, для того чтобы можно было его изменять.

T & operator[](size\_t index);

Второе определение возвращает новый объект, для случая, когда мы работаем с контейнером, который нельзя менять.

T operator[](size\_t index);

1. Дайте определение вложенного класса.

Класс внутри класса(and we need to go deeper :) )

1. Можно ли класс-итератор реализовать как внешний класс? А как вложенный? В чем отличия этих методов реализации?
2. Может ли объемлющий класс иметь неограниченный доступ к элементам вложенного класса? А вложенный класс — к элементам объемлющего?

Я пока не очень понимаю, как иметь допуск не к static элементам. Надо получше про это узнать.

1. Ограничена ли глубина вложенности классов?

 ?? Где про это можно узнать?

1. Можно ли определить вложенный класс внешним образом? Зачем это может понадобиться?

Да. Зачем - непонятно.

1. Каким образом вложенный класс может использовать методы объемлющего класса? А объемлющий — методы вложенного?

Я пока не очень понимаю, как иметь допуск не к static элементам. Надо получше про это узнать.

1. Что такое «запредельный» элемент, какую роль он играет в контейнерах?

В стандартных контейнерах это end(), используется для обозначения конца контейнера, возвращения в случае неудачного поиска, и т.д.

1. Объясните, по каким причинам трудно написать универсальный контейнер, элементы которого могут иметь произвольный тип.