1. Когда используются контейнеры типа (мульти) множество и отображение?

Стоит отметить, что множества и мультимножества упорядочивают свои элементы в соответствии с заданным критерием, разница между ними заключается в том, что мультимножества допускают дубликаты, а множества нет. Важное замечание: мультимножества обычно реализуются в виде сбалансированных бинарных деревьев, так же критерий сортировки должен определять строгое слабое упорядочивание

Контейнеры типа мультимножества эффективны при сортировке, т.к основное преимущество автоматической сортировки состоит в том, что бинарное дерево допускает эффективный поиск конкретного значения(имеет логарифмическую сложность)

Отображение и мультиотображения – контейнеры, элементами которого являются пары <ключ - значение>. Эти контейнеры автоматически упорядочивают свои элементы в соответствии с определенным критерием сортировки, заданным для ключа. Разница отображения и мультиотображения состоит в том, что при мульти допустимы дубликаты.

Как и все ассоциативные контейнерные классы, отображение и мультиотображение обычно реализуются в виде сбалансированных бинарных деревьев, важно сказть, что ключ и значение – один и тот же объект. Таким образом, отображение и мультиотображение имеют все возможности и операции, что и множества и мультимножества. Однако есть различия, во-первых, их элементами являются пары <ключ-значение>, во –вторых, отображения можно использовать как ассоциативные массивы.

Отображение и мульти автоматически упорядочивают свои элементы п их ключам, это обеспечивает хорошее быстродействие при поиске элементов по заданному ключу.

1. Каким требованиям должна удовлетворять качественная хэш-функция?

Unordered-контейнеры

Требования:

* Детерминированность
* Скорость вычисления хеша(не должна зависеть от количества элементов, хранящихся в хеш таблице, должна зависеть от объекта(длина объекта))
* Равномерность

1. Из-за чего в хэш-таблицах возникают коллизии и как можно их разрешать?

Первое, что такое хэш таблица? - это контейнер Принцип действия: мы определяем функцию хеширования, которая по каждому входящему элементу будет определять натуральное число. А уже дальше по этому натуральному числу мы будем класть элемент в (допустим) массив. Тогда имея такую функцию мы можем за O(1) обработать элемент.

Коллизии возникают в хеш-таблицах из-за совпадения, способы борьбы с коллизией - метод цепочек или открытая адресация(мы будем спускаться до того момента, пока не дойдем до пустой ячейки, в эту ячейку мы и загрузим значение)

1. Почему сложность основных операций хэш-таблиц в худшем случае O(N)?

На специальных данных асимптотика может вырождаться в O(N), на скорость работы могут влиять удаленные элементы, т.к с ними мы ничего сделать не можем

1. Что позволяет сделать инструмент создания контейнеров Boost Multiindex?

Всё усложняется когда возникает необходимость в классе для хранения и доступа к объектам по более чем одному ключу или их комбинациям, тогда и используется boost.multiindex

Boost.Multiindex – средство для создания контейнеров с множественными интерфейсами (vector, set, hash и тд)