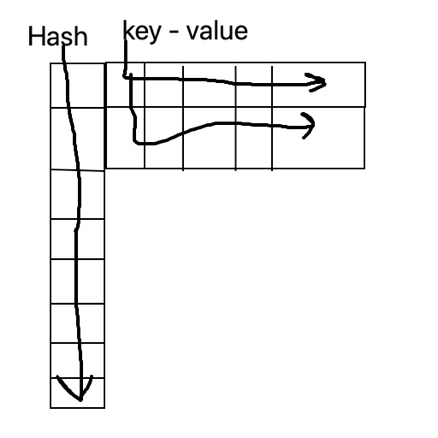
Короче говоря я файлик для сдачи лабы по хеш-таблицам.

Так смысл каждой задачи написать функции для каждого вида хеш-таблиц.

Плюс нам нужно разработать структуру для каждой задачи.

Задача 1:

Мы создаём вектор в векторе, то есть каждая ячейка в векторе будет являться вектором. Это нам нужно для эффективного использования хеш-таблицы.

Вот так оно примерно выглядит. То есть, когда мы выполняем процедуру вставки мы с помощью хеш-функции (это такая херня, которая преобразует наша число или строка в какое-то ёбнутое число). Мы хешируем элемент после чего в начало вектора вставляем хеш-код, а после пару ключ-значение. Удаление и поиск также сначала хешируют входящий элемент после ищут его и выводят если поиск или удаляют если это функция удаления.

Задача 2.

Смысл данной задачи создать ассоциативный массив. У вас мог возникнуть вопрос: А что это за поебень такая? А я вам отвечу тут ничего сложного нет. Обычный массив — это частный случай ассоциативного. Обычный массив хранит ключ–значение, где ключ по дефолту это индекс 0, 1, 2, … , n. А ассоциативный массив самому указать чем будут являться наши ключи. В нашем случае это хеш код ключа.

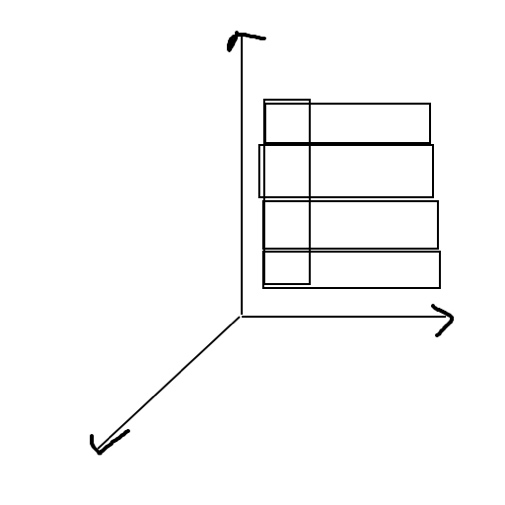
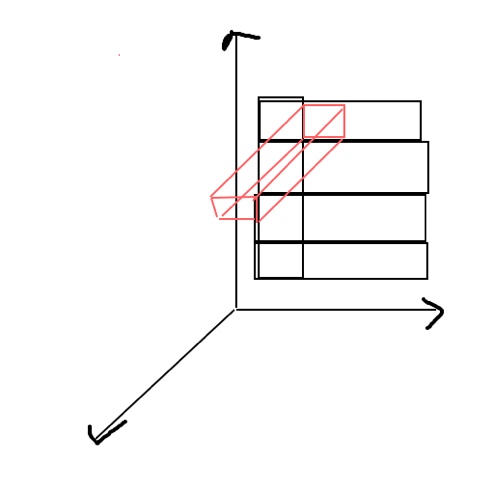
Тут в принципе всё то же самое, что и в первой задаче только теперь мы создаём вектор не для целых чисел, а с помощью структуры для хранения строк ( vector< vector<Key> > hash\_table(n, vector<Key>()); ). Все функции в принципе делают всё то же самое поэтому ебись это задача конём.

Задача 3.

Тут уже мои полномочия всё, пошла залупа, скорее всего эту задачу вы, мой дорогой читатель, не сдадите Кравцову… Что же тут нахуй происходит: у нас есть структура для каждого элемент которая будет хранить ключ – значение, указатель на следящий элемент в цепи, указатель на предыдущий элемент в цепи, и указатель на следящую цепь. Кста мой рисунок к первой задачи по-прежнему актуален, за исключением того, что у нас есть связи между текущим, следующим и предыдущим. Помимо всей этой ебанины у нас есть два класса: первый осуществляет все основные функции, а второй оптимизирует его для работы с ассоциативным массивом. Там в каждой функции дохуя нюансов, но молите Величайшего, чтобы он не взошёл на вас и сидел с вами на защите как ангел-хранитель и нашёптывал вам ответы на вопросы практика. Увсё.

Задача 4.

Если вы думали, что до этого была какая-то хуйня и вы ничего не поняли, то щас вы вообще кончите. Мултимап что это за покемон и как его ебать.

Это тип как на пикчах было и стало. У нас была двухмерная таблица, а тут вдруг ебаться в телевизор и стала трёхмерная. В чём суть: теперь у нас есть вектор для хешей из него идёт вектор для ключей, а из него вектор для значений. Это как раз и позволяет нам храть несколько значений для нашего ключа. Но нахуй не тут-то было на практике у нас четырёхмерный вектор. Ебать прикол да. Потому что нам нужно будет хранить вектор для хешей от значений. То есть у нас идёт поиск хеша для ключа, после поиск самого ключа, после поиск хеша для значения и собственно нахождения искомого значения. Вот такая вот ебанина ребята. Если говорить по фактам, мультиМап это хеш-таблица в хеш-таблице. Большое спасибо, что прочитали отрывок из моей книги: “Я сосал, меня ебали”. Подробности смотрите в источнике: https://www.гейчик.ком