1. Если мы хотим реализовать поиск за постоянное время, т.е. О(1).

Время поиска О(1) дают хеш-таблицы, так что будем рассматривать хранение данных в хеш-таблице. В хеш-таблице данные хранятся в формате ключ-значение. Пусть на множестве ключей определена какая-то хеш-функция и пусть хеш занимает 16 бит. В лучшем случае, если мы хотим константное время поиска, нам потребуется 2^16 значений. Но в реальности так никто не делает, так как в хеш-таблицах существует понятие соотношение занятых элементов к общему объему (сейчас в наших рассуждениях общий объем 2^16, а занято – 18758328). В C++, например, при создании хеш-таблицы она создается не на все 2^16 значений, а сначала поменьше, чтобы постоянно было занято, к примеру 75%. Допустим, что в нашей задаче тоже возьмем постоянный занятый объем памяти 75%.

(18758328 / 75 \* 100) \* 20 символов \* 16 бит (длина каждого символа, если мы предполагаем возможность использования и латиницы и кирилицы) = 8 \* 10^9.

1. Если мы хотим занять как можно меньше памяти.

Если выбрать такую структуру хранения, как сортированный массив, то размер памяти будет минимальный. Общий занимаемый объём будет рассчитываться, как размер массива (18758328 элементов) \* размер элемента массива (20 символов \* 16 бит). Всего: 6 \* 10^9.