Лабораторна робота №2 Хеш таблиці

**Хеш-табли́ца** — это [структура данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85" \o "Структура данных), реализующая интерфейс [ассоциативного массива](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2" \o "Ассоциативный массив), а именно, она позволяет хранить пары (ключ, значение) и выполнять три операции: операцию добавления новой пары, операцию поиска и операцию удаления пары по ключу.

Существуют два основных варианта хеш-таблиц: с цепочками и открытой адресацией. Хеш-таблица содержит некоторый [массив](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) {\displaystyle H}, элементы которого есть пары (хеш-таблица с открытой адресацией) или списки пар (хеш-таблица с цепочками).

Выполнение операции в хеш-таблице начинается с вычисления [хеш-функции](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) от ключа. Получающееся хеш-значение {\displaystyle i={\mbox{hash}}(key)} играет роль индекса в массиве {\displaystyle H}. Затем выполняемая операция (добавление, удаление или поиск) перенаправляется объекту, который хранится в соответствующей ячейке массива {\displaystyle H[i]}.

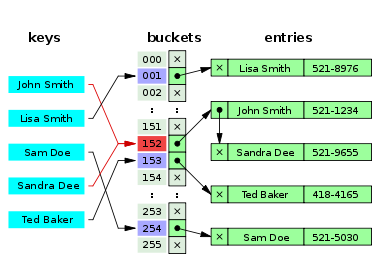
Ситуация, когда для различных ключей получается одно и то же хеш-значение, называется [коллизией](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8)

Складність пошуку в ідеальній хеш-таблиці O(1), в найгіршому випадку O(n)

Разрешение коллизий[[править](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&veaction=edit&section=3) | [править код](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&action=edit&section=3)]

Существует несколько способов разрешения [коллизий](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8).

**Метод цепочек**[[править](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&veaction=edit&section=4) | [править код](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&action=edit&section=4)]

[](https://commons.wikimedia.org/wiki/File:Hash_table_5_0_1_1_1_1_1_LL.svg?uselang=ru)

Разрешение коллизий при помощи цепочек.

Каждая ячейка массива *H* является указателем на [связный список](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%8F%D0%B7%D0%BD%D1%8B%D0%B9_%D1%81%D0%BF%D0%B8%D1%81%D0%BE%D0%BA) (цепочку) пар ключ-значение, соответствующих одному и тому же хеш-значению ключа. Коллизии просто приводят к тому, что появляются цепочки длиной более одного элемента.

Операции поиска или удаления элемента требуют просмотра всех элементов соответствующей ему цепочки, чтобы найти в ней элемент с заданным ключом. Для добавления элемента нужно добавить элемент в конец или начало соответствующего списка и в случае, если коэффициент заполнения станет слишком велик, увеличить размер массива *H* и перестроить таблицу.

При предположении, что каждый элемент может попасть в любую позицию таблицы *H* с равной вероятностью и независимо от того, куда попал любой другой элемент, среднее [время работы](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0) операции поиска элемента составляет [Θ](https://ru.wikipedia.org/wiki/%D0%9E-%D0%BD%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D1%8F)(1 + *α*), где *α* — коэффициент заполнения таблицы.

**Открытая адресация**[[править](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&veaction=edit&section=5) | [править код](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&action=edit&section=5)]

[](https://commons.wikimedia.org/wiki/File:Hash_table_5_0_1_1_1_1_0_SP.svg?uselang=ru)

Пример хеш-таблицы с открытой адресацией и линейным пробированием, получающейся при вставке элементов в левой колонке сверху вниз.

В массиве *H* хранятся сами пары ключ-значение. Алгоритм вставки элемента проверяет ячейки массива *H* в некотором порядке до тех пор, пока не будет найдена первая свободная ячейка, в которую и будет записан новый элемент. Этот порядок вычисляется на лету, что позволяет сэкономить на памяти для указателей, требующихся в хеш-таблицах с цепочками.

Последовательность, в которой просматриваются ячейки хеш-таблицы, называется **последовательностью проб**. В общем случае, она зависит только от ключа элемента, то есть это последовательность *h*0(*x*), *h*1(*x*), …, *hn*— 1(*x*), где *x* — ключ элемента, а *hi*(*x*) — произвольные функции, сопоставляющие каждому ключу ячейку в хеш-таблице. Первый элемент в последовательности, как правило, равен значению некоторой хеш-функции от ключа, а остальные считаются от него одним из приведённых ниже способов. Для успешной работы алгоритмов поиска последовательность проб должна быть такой, чтобы все ячейки хеш-таблицы оказались просмотренными ровно по одному разу.

Алгоритм поиска просматривает ячейки хеш-таблицы в том же самом порядке, что и при вставке, до тех пор, пока не найдется либо элемент с искомым ключом, либо свободная ячейка (что означает отсутствие элемента в хеш-таблице).

Удаление элементов в такой схеме несколько затруднено. Обычно поступают так: заводят булевый флаг для каждой ячейки, помечающий, удален элемент в ней или нет. Тогда удаление элемента состоит в установке этого флага для соответствующей ячейки хеш-таблицы, но при этом необходимо модифицировать процедуру поиска существующего элемента так, чтобы она считала удалённые ячейки занятыми, а процедуру добавления — чтобы она их считала свободными и сбрасывала значение флага при добавлении.

Я замість цього флага створив особливий елемент nool, це змешить розміри таблиці.

Моя хеш функція f(key)=key % 1000 000

В моїй хеш таблиці з відкритою адресацією, якщо місце було зайняте: n(next)=(n+1) % 1 000 000

Проги:

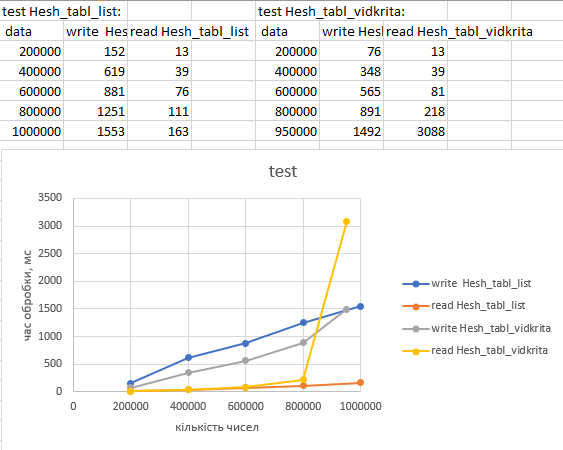
Хеш таблиця зі списками lab\_2\_hesh\_list\ lab\_2\_hesh\_list.sln

Хеш таблиця з відкритою адресацією lab\_2\_hesh\_vidkrita\ lab\_2\_hesh\_vidkrita.sln

Файл з статистикою читання і запису різної кількості даних: lab\_2\_hesh… \bin\Debug\data1.csv

Файл з невеликою кількістю тестових даних: lab\_2\_hesh… \bin\Debug\data2.csv

Графіки: data2.xlsx



Висновок: в хеш таблиці зі списками час запису (читання) майже лінійний від кількості записаних(зчитаних) елементів. В хеш таблиці з відкритою адресацією час читання і запису спочатку майже лінійний, але коли таблиця заповняється більше ніж на 80% час запису і пошуку прямує від лінійного до квадратичного (бо для пошуку місця вставки і пошуку елементу доводиться перебирати майже всю таблицю).

**Назви хеш таблиць не співпадають з лекцією, бо в лекції термінологія не відповідає міжнародній (хтось в лекції переплутав назви таблиць).**