Java Basic

lecture #15. Hashing, Hash Function and Hash Table

Mentor: Il'yas Miftakhov

lecture #15. Hashing, Hash Function and Hash Table

- Hashing Introduction
- Using Hashing
- Basic operations
- Collisions
- Easy Practice

Что такое Хэш (Hash) ?

Hashing - это преобразование любого объема информации в уникальный набор символов, который относится только к этому массиву входящей информации.
Этот набор символов и будет называться хэшем.

И еще раз:
Хэш или хэш-функция – это математический алгоритм, преобразовывающий произвольный массив данных в состоящую из букв и цифр строку фиксированной длины.

Пример: одна из самых распространенных hash-function преобразует мое имя Il'yas в hash

b55bf6955748e4699b60b4c519bd710688a8f310

Попробуйте и вы свое имя ☺
http://www.sha1-online.com/
Попробуйте со строчной буквы
Попробуйте получить hash из "My name is <your name> "

Hashing

Обязательные свойства Hash:

- Хэш всегда уникален для каждого массива информации, иногда случаются коллизии, когда для разных входных блоков информации вычисляются одинаковые хэш-коды.
- При самом незначительном изменении входной информации - хэш полностью меняется.
- Хэш-функция необратима и не позволяет восстанавливать исходный массив информации из символьной строки.
- Хэширование позволяет достаточно быстро вычислить нужный хэш для достаточно большого объема информации.
- Алгоритм работы хэш-функции, как правило, делается открытым, чтобы при необходимости можно было оценить ее стойкость к восстановлению начальных данных по выдаваемому хэшу.
- Хэш-функция должна уметь приводить любой объем данных к числу заданной длины.

Использование Hash

1. Работа с большими объемами информации
2. Проверка целостности данных при передаче
3. Шифрование
4. Электронные цифровые подписи
5. Хранение паролей

Hash или Hash-function – одна из основных составляющих современной криптографии и алгоритмов блокчейна.

Basic operation

- HashTable: операция используется для создания новой хеш-таблицы.
- Delete: операция используется для удаления определенной пары ключ-значение из хеш-таблицы.
- Get: операция используется для поиска ключа внутри хеш-таблицы и возврата значения, связанного с этим ключом.
- Put: операция используется для вставки новой пары ключ-значение в хеш-таблицу.
- DeleteHashTable: операция используется для удаления хеш-таблицы.

О коллизиях (конфликтах)

- SHA-256
- 256 бит - это 2^256 соответствий, то есть 2^256 различных входов имеют свой уникальный хеш.

коллизия происходит, когда разные входные данные производят одинаковый хеш

Хеш-функция считается устойчивой к коллизиям до того момента, пока не будет обнаружена пара сообщений, дающая одинаковый выход.

Хеш-функция считается устойчивой к коллизиям, когда вероятность обнаружения коллизии настолько мала, что для этого потребуются миллионы лет вычислений.

Хеш-функций без коллизий не существует ☺

Как обрабатывать коллизии?

Есть два основных метода обработки коллизий:

1. Раздельная цепь (Separate Chaining)
2. Открытая адресация (Open Addressing)

Для метода 1 - Идея состоит в том, чтобы реализовать массив в виде списка, называемого цепью.

Для метода 2 – Идея состоит в том, что используем следующий пустой слот в самой хэш-таблице.

Практика

1. Контракт hash-code и equals
2. Первый элемент, встречающийся k раз в массиве (easy)
3. Проверьте, равны ли два массива или нет (easy)
4. Сгруппируйте слова с одинаковым набором символов (middle)

SOLID

S

Принцип единственной ответственности (single responsibility principle)

Для каждого класса должно быть определено единственное назначение. Все ресурсы, необходимые для его осуществления, должны быть инкапсулированы в этот класс и подчинены только этой задаче.

O

Принцип открытости/закрытости (open-closed principle)

«программные сущности … должны быть открыты для расширения, но закрыты для модификации».

L

Принцип подстановки Лисков (Liskov substitution principle)

«функции, которые используют базовый тип, должны иметь возможность использовать подтипы базового типа не зная об этом». См. также контрактное программирование.

I

Принцип разделения интерфейса (interface segregation principle)

«много интерфейсов, специально предназначенных для клиентов, лучше, чем один интерфейс общего назначения»

D

Принцип инверсии зависимостей (dependency inversion principle)

«Зависимость на Абстракциях. Нет зависимости на что-то конкретное»