Dictionaries, Maps and Sets

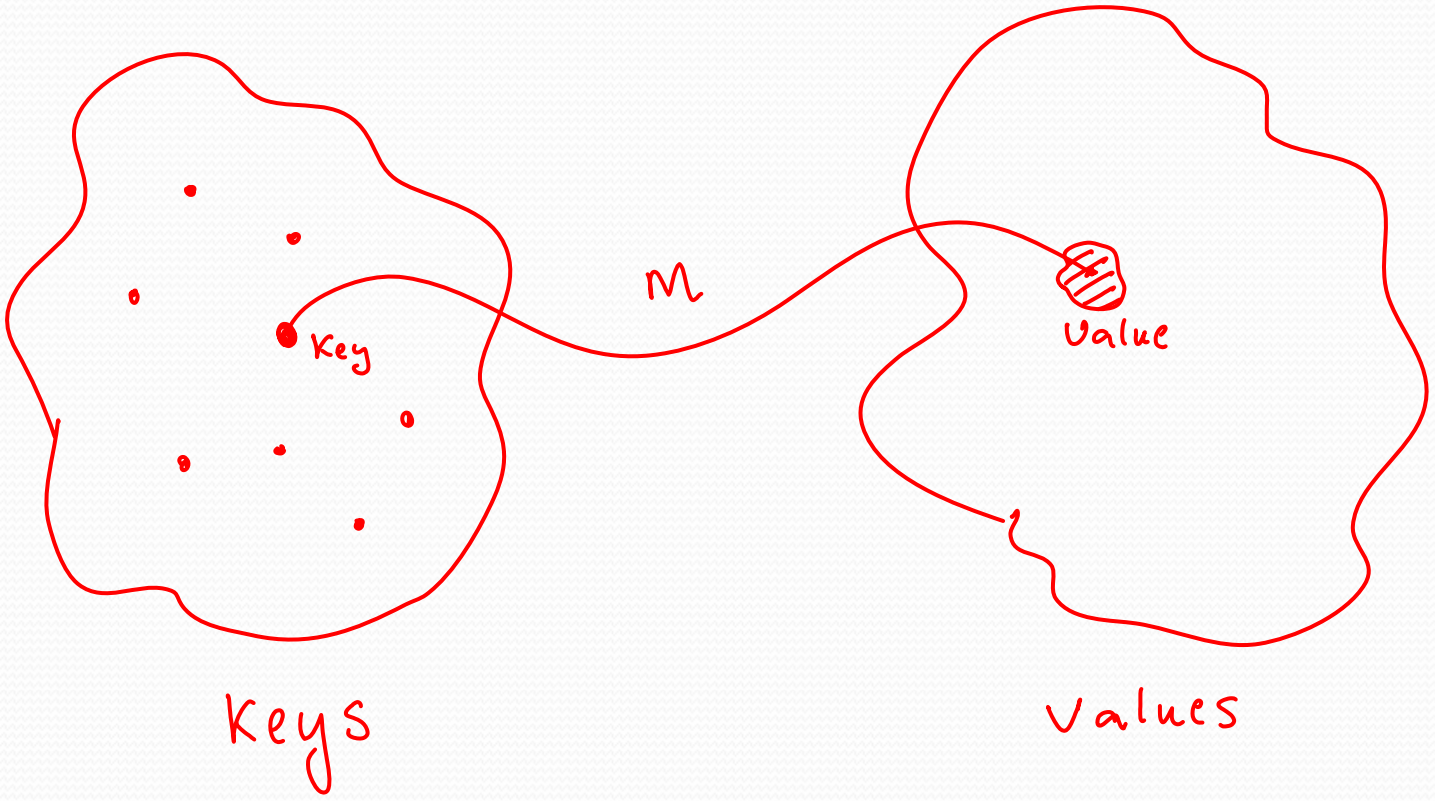
15-121 Introduction to Data Structures

Ananda Gunawardena

Maps

Maps

Map is a relation between set of keys and set of values

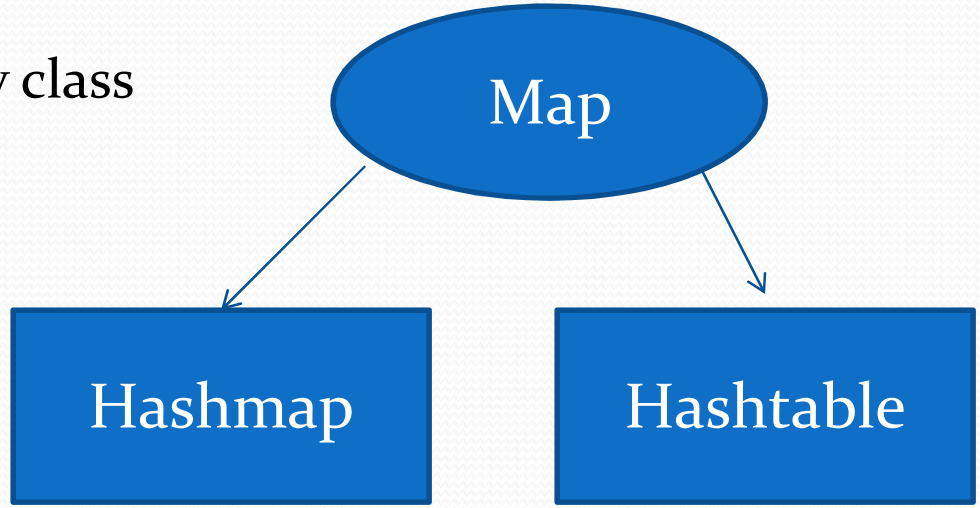


Maps

- An object that maps keys to values.
- A map cannot contain duplicate keys
- Each key can map to at most one value.
- Java Map interface can be implemented by any class
  - **get**(Object key)
  - **keySet**()
  - **put**(Object key, Object value)

Map ADT

- Java provides a Map interface
- Some implementing classes
  - Hashmap
    - Hashtable implementation of a map interface
  - Hashtable
    - Extended dictionary class



Example of Maps

- Taurus;Kesden, Greg
- RX-8;Slater, Don
- Accord;Mouse, Mickey
- Camry;Roberts, Jim
- Dakota;Pattis, Rich
- F-150;Stehlik, Mark
- I-300;McElfresh, Scott
- big yellow bus;Cortina, Tom

Dictionary Data Structure aka hash tables

- Dictionary is a data structure that supports lookups and updates efficiently

- Computational complexity of Dictionary:
  - The time and space needed to authenticate the dictionary, i.e. creating and updating it.
  - The time needed to perform an authenticated membership query.
  - The time needed to verify the answer to an authenticated membership query.

Java Dictionary Class - abstract

- The Dictionary class is the abstract parent of Hashtable, which maps keys to values.
- A Dictionary object, every key is associated with at most one value
- Given a Dictionary and a key, the associated element can be looked up
- Any non-null object can be used as a key and as a value.
- Some Methods :
  - **get**(Object key)
  - **put**(Object key, Object value)
  - **remove**(Object key)

Java Tools - HashMap Class

- This implements the Map interface
- HashMap permits null values and null keys.
- Constant time performance for get and put operations
- HashMap has two parameters that affect its performance: *initial capacity* and *load factor*
  - Capacity – number of buckets in the Hash Table
  - Load factor – How full the Hash Table is allowed to get before capacity is automatically increased using rehash function.

HashMap API

java.util

Class HashMap

java.lang.Object
  └ java.util.AbstractMap
    └ java.util.HashMap

All Implemented Interfaces:

Cloneable, Map, Serializable

Direct Known Subclasses:

LinkedHashMap, PrinterStateReasons

---

public class **HashMap**
extends AbstractMap
implements Map, Cloneable, Serializable

Hash table based implementation of the `Map` interface. This implementation provides all of the optional map operations, and permits `null` values and the `null` k
(The `HashMap` class is roughly equivalent to `Hashtable`, except that it is unsynchronized and permits nulls.) This class makes no guarantees as to the order of t
map; in particular, it does not guarantee that the order will remain constant over time.

More on Java HashMaps

- HashMap(int initialCapacity)
- put(Object key, Object value)
- get(Object key) – returns Object
- keySet() Returns a **set** view of the keys contained in this map.
- values() Returns a **collection** view of the values contained in this map.

Sets

Sets

- A set is defined as a collection that contains no duplicates
- Java Set interface provides number of methods. Among some of them are
  - **add**(Object element)
    - Adds the specified element to this set if it is not already present
  - **contains**(Object element)
    - Returns true if this set contains the specified element.

Classes that implements set interface

- public class HashSet
  - Implements Set interface using HashTable
  - Offers constant time performance
- public class TreeSet
  - Implements Set interface using TreeMap
  - Sorted set will be ascending order according to the natural order

Exercise

- Write a method
  - public static Set intersect(Set A, Set B)
- That takes two sets A and B and return the intersection of the two sets

- Write a method
  - public static Set union(Set A, Set B)
- That takes two sets A and B and return the union of the two sets