Maps

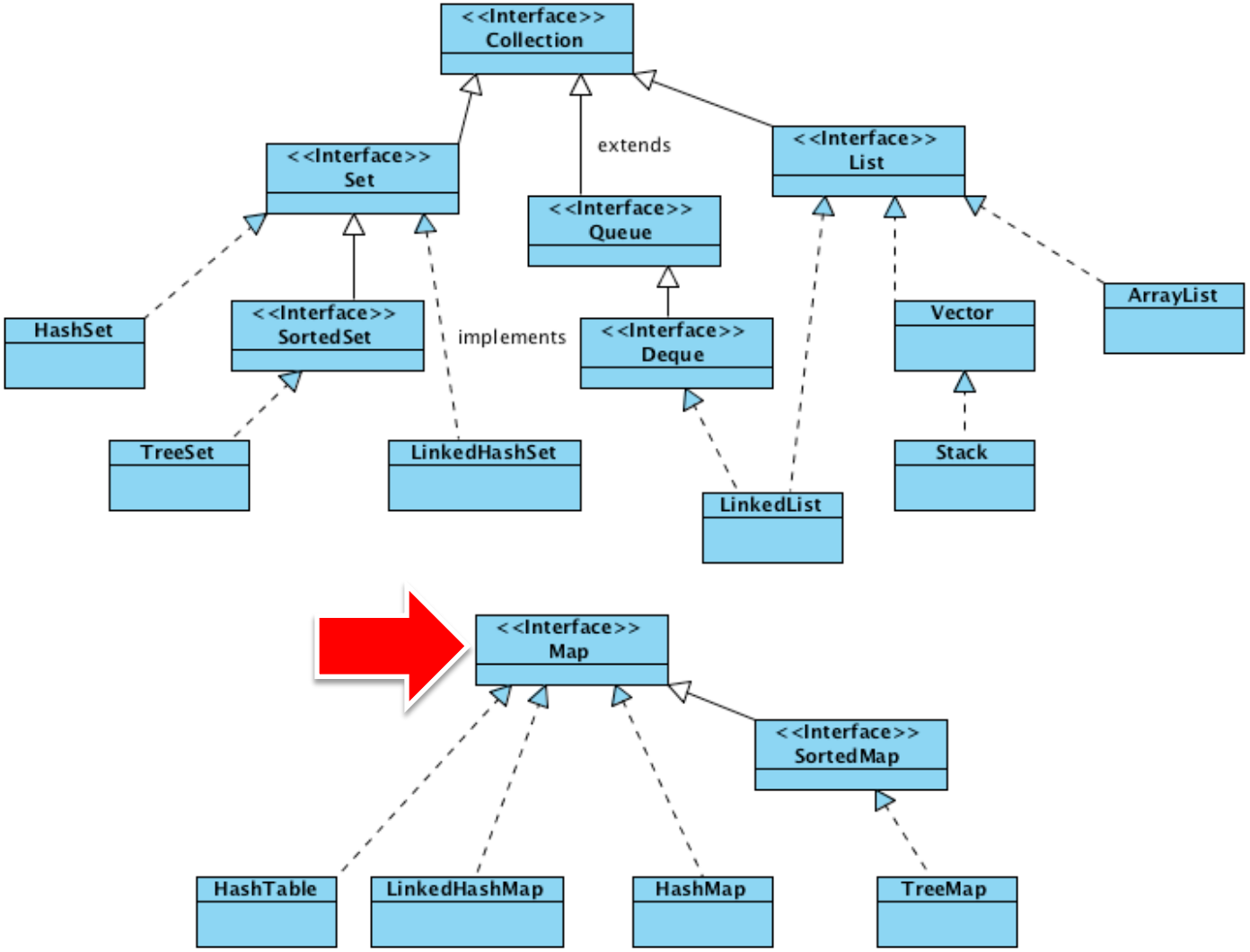
a.k.a, associative array, map, or dictionary

Definition

- In computer science, an **associative array**, **map**, or **dictionary** is an abstract data type composed of (key, value) pairs, such that each key appears at most once
- Modern programming languages natively supports them E.g. Perl, Python, Ruby, Go
- Implemented through hash tables or tree data structure

```
V1[42] = "h2g2"
V2["h2g2"] = 42
```

Java Collection Framework



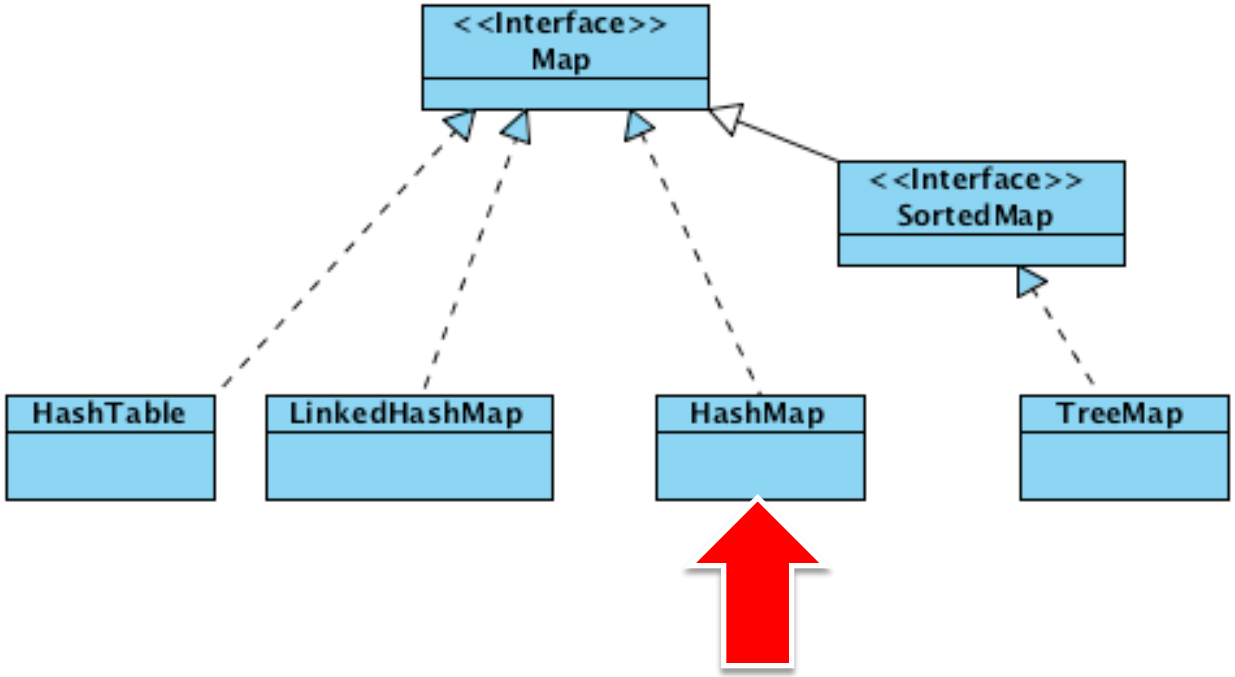
Map interface

- Map<K,V>
  - K: the type of keys maintained by this map
  - V: the type of mapped values
- Add/remove elements
  - value **put**(key, value)
  - value **remove**(key)
- Search
  - boolean **containsKey**(key)
  - boolean **containsValue**(value)

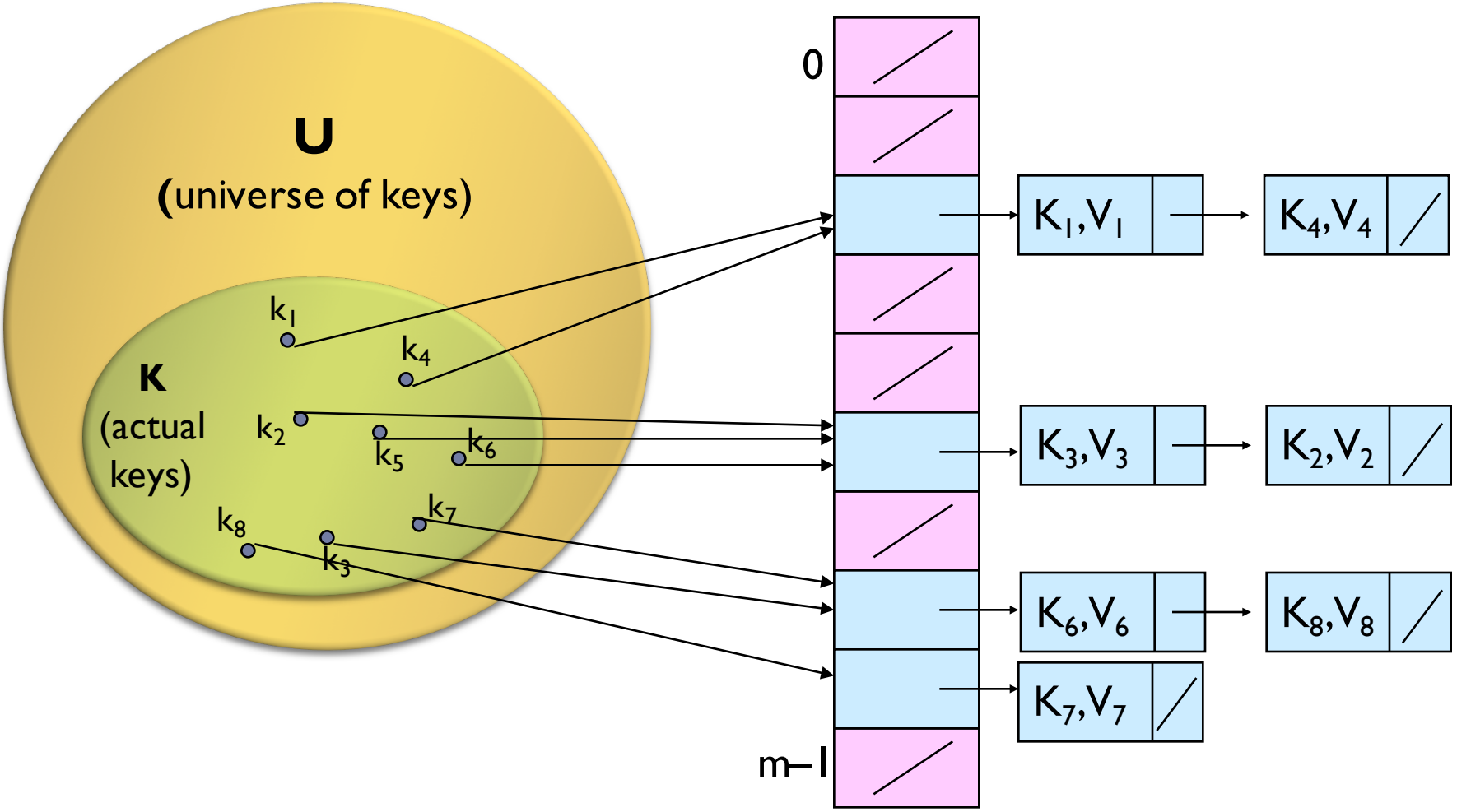
Map interface (cont.)

- Nested Class
  - Map.Entry<K,V>
  - A map entry (key-value pair).
- Set<Map.Entry<K,V>> entrySet()
  - Returns a **Set view** of the mappings contained in this map
- Set<K> keySet()
  - Returns a **Set view** of the keys contained in this map
- Collection<V> values()
  - Returns a **Collection** view of the values contained in this map

Map Family Tree



HashMap and Chaining



HashMap and Chaining

- Non duplicated keys (values could be duplicated)
  - Chaining is not used to store multiple keys with the same value. Each key should be unique
  - Chaining is used to solve the collision problem.

HashMap

- Non duplicated keys (values could be duplicated)
- Not ordered (neither sorted)

- Implementation is based on a hash table
  - Operations *put(k, v)*, *get(k)*, *remove(k)*, *containsKey(k)* have complexity mostly O(1)

- Requires to override hashCode() equals()
- Key object must be immutable

HashMap vs HashSet

- HashMap allows to insert key-value pairs. Each key is associated to a value

- HashSet allows to insert an object in a collection of object. The object itself (or part of it) is the key

- Similarties:
  - Do not accept duplicated key
  - Not ordered (neither sorted)
  - Implementation is based on a hash table
  - Requires to override hashCode() equals() for the Key object
  - Key object must me immutable (at least for the field used in hashCode() and equals())

HashMap complexity

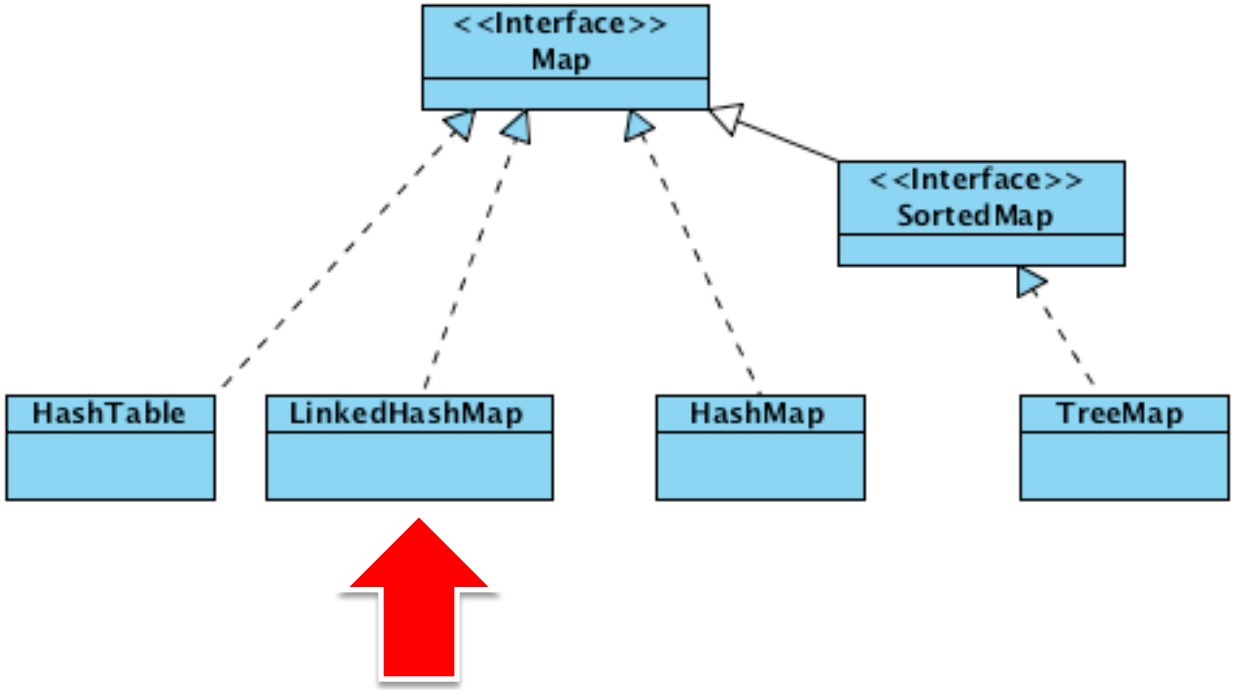
	HashMap
put(key, object)	O(1)
get(key)	O(1)
remove(key)	O(1)
containsKey(key)	O(1)
containsValue(object)	O(N)

HashMap complexity

containsValue() will probably require time *linear in the map size* for most implementations of the Map interface – i.e. it is O(N)

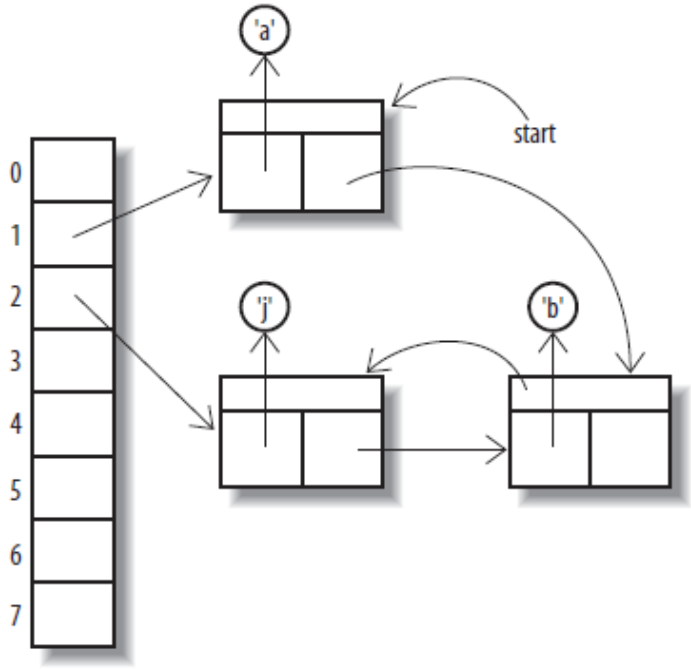
put(key, object)	
get(key)	O(1)
remove(key)	O(1)
containsKey(key)	O(1)
containsValue(object)	O(N)

Collection Family Tree

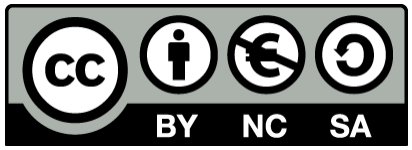


LinkedHashMap

- Implementation is based on a hash table and a double-linked list running through all of its entries:
  - Operations *put(k, v)*, *get(k)*, *remove(k)*, *containsKey(k)* have complexity mostly O(1)
- Non duplicated keys
  - Values could be
- Ordered (usually insertion-order)
  - Insertion order is not affected if a key is re-inserted
- Not sorted



Licenza d'uso



- Queste diapositive sono distribuite con licenza Creative Commons "Attribuzione - Non commerciale - Condividi allo stesso modo (CC BY-NC-SA)"
- Sei libero:
  - di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
  - di modificare quest'opera

- Alle seguenti condizioni:
  - **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo i cui tu usi l'opera.

  - **Non commerciale** — Non puoi usare quest'opera per fini commerciali.

  - **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con un licenza identica o equivalente a questa.

- http://creativecommons.org/licenses/by-nc-sa/3.0/