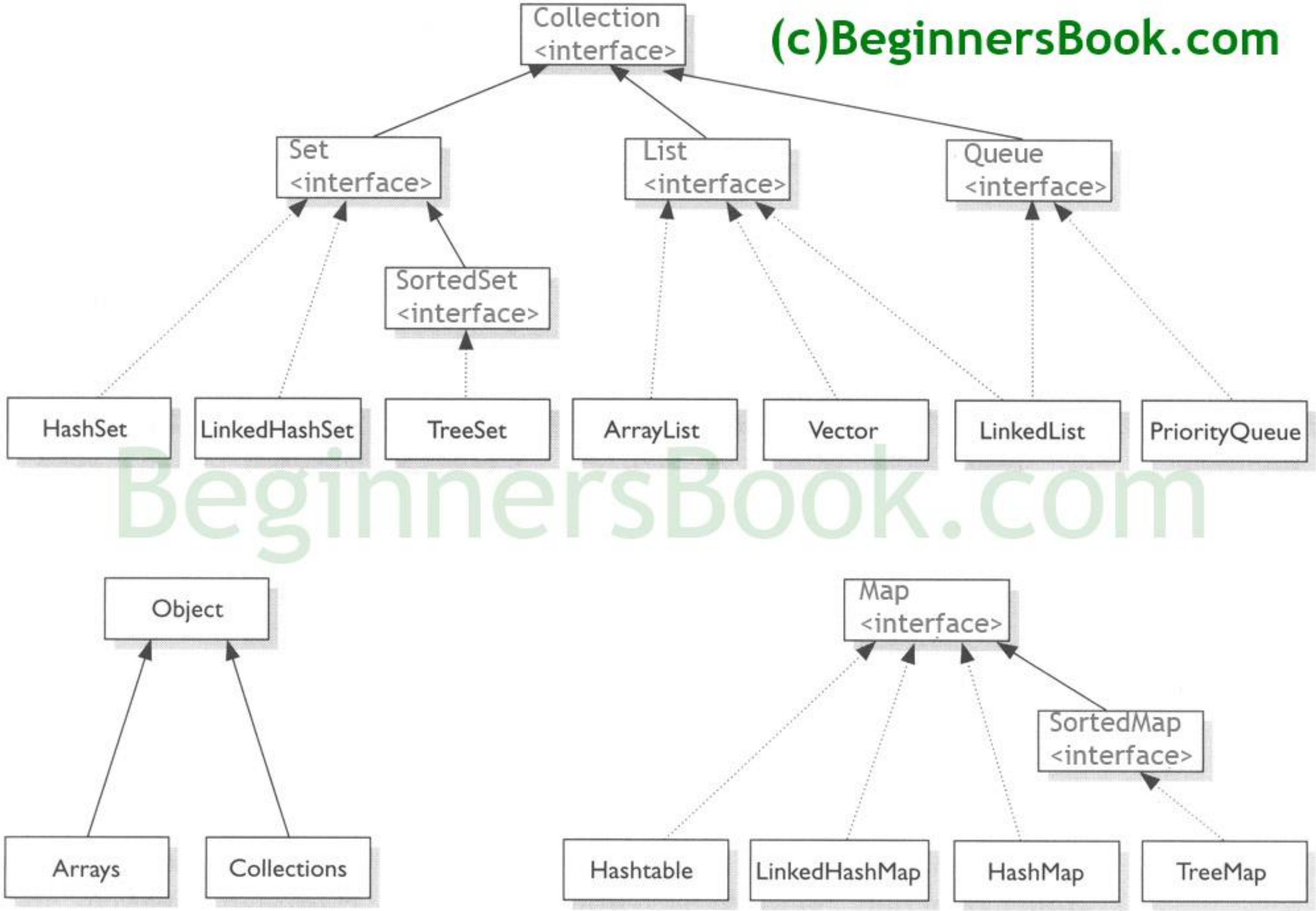
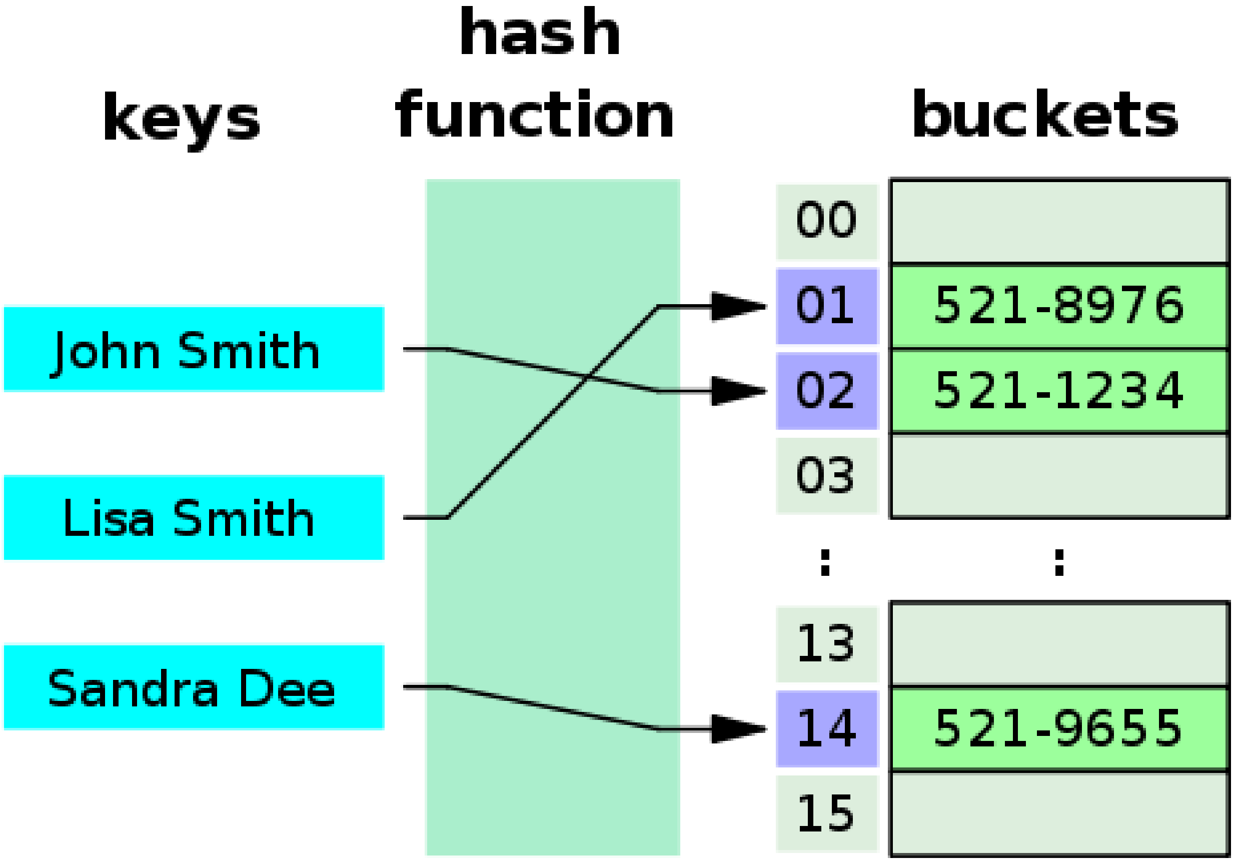


All interfaces are marked with <interface> tag in the diagram. Other boxes represent classes.

implements

extends

Hash table


keys
hash function
buckets
John Smith
Lisa Smith
Sandra Dee
00
01
02
03
:
13
14
15
521-8976
521-1234
:
521-9655

Hashing – principle

- Data is stored in an **array**
- The index **is calculated** based on a key.
  - **index = hash-function (key)**
- Insert
  - put(key, value)
- ”Search”
  - get (key)
- The **hash-function** must
  - return an integer (index < size of table)
  - be easy to calculate (why?)
  - Minimize the number of collisions
  - distribute data elements evenly across the table (why?)

Hash function (ex):

key value modulo 11

(11 = size of table)

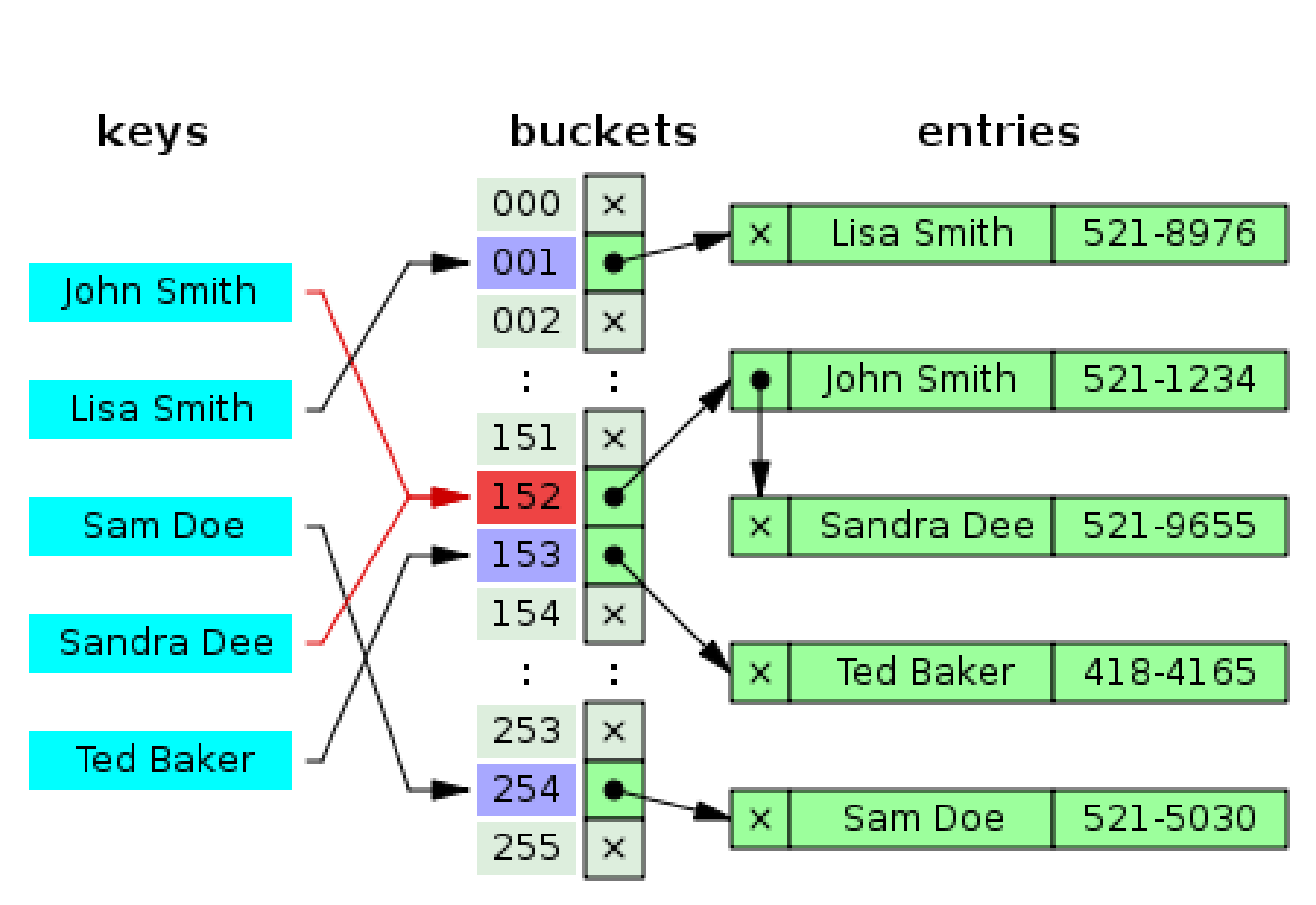
ex:

Key value: 13

hash (13) => 13 mod 11 => 2

0	11
1	1
2	13
3	
4	
5	
6	
7	
8	30
9	
10	10

Collisions, chaining



Hash collision resolved by separate chaining.

Efficiency and hash table

- Insert, delete and search is (nearly) independent of the number of elements (n)
  - O(1)
  - Load factor

- Table size << number of possible different key values

- Preferred when you require fast
  - search
  - Insert

  but not fast
  - Iterate sorted
  - Location of max /min