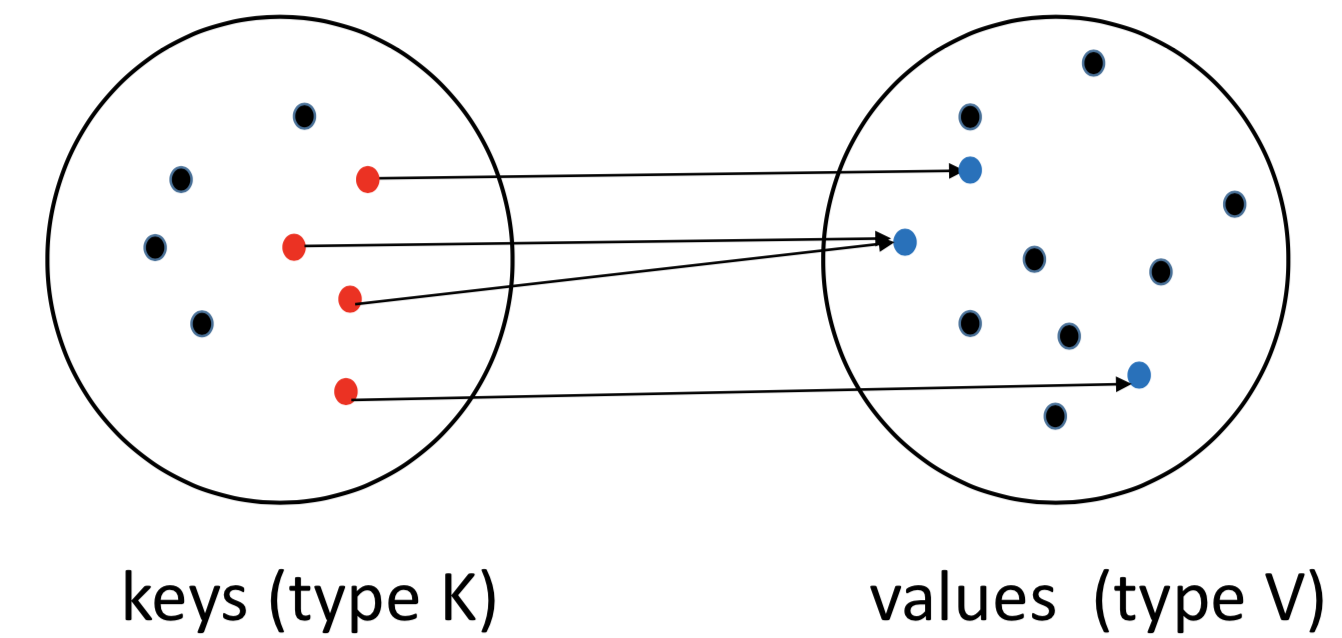
COMP250-Map

*A map is a set of (key, value) pairs.*

*For each key, there is at most one value.*



The black dots here indicate keys or values that are not in the map.

*Each (key, value) pair is called an entry. In this example, there are four entries.*

* **put( key, value )**

If the map previously contained a mapping for the key, the old value is replaced by the specified value, and previous value is returned. Otherwise, return null

can null be key or value? (definition)

* **get(key)**

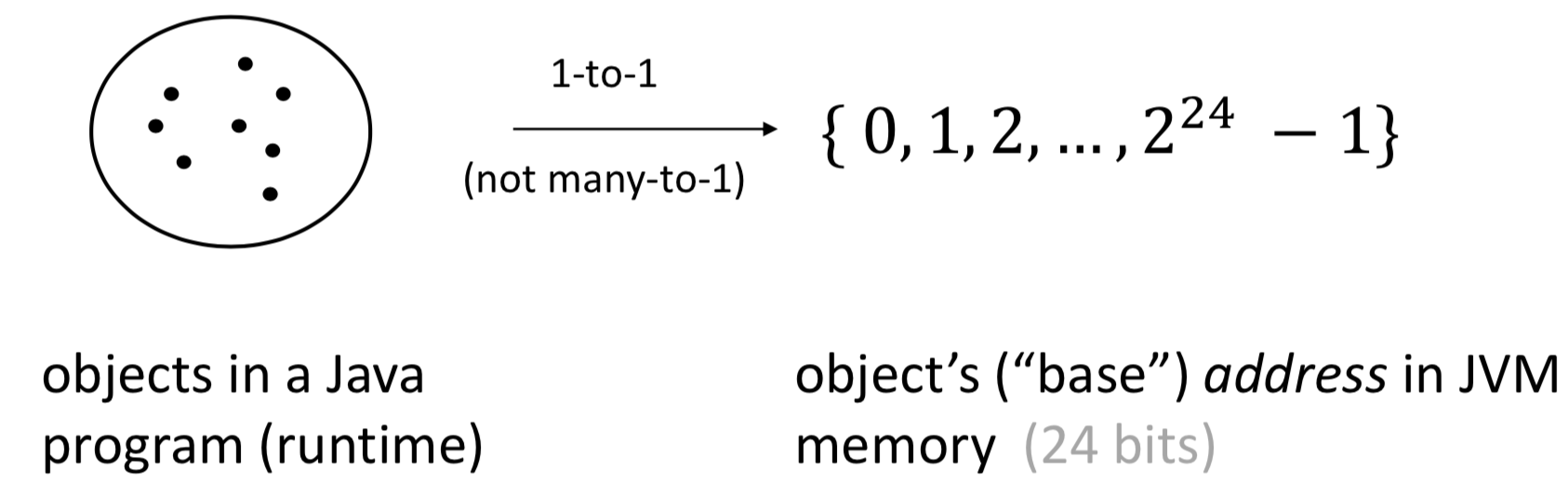
Returns the value to which the specified key is mapped, or null if this map contains no entry for the key.

* **remove(key)**

Removes the entry for the key, if it is present, and returns the value to which this map previously associated the key, or null if the map contained no mapping for the key.

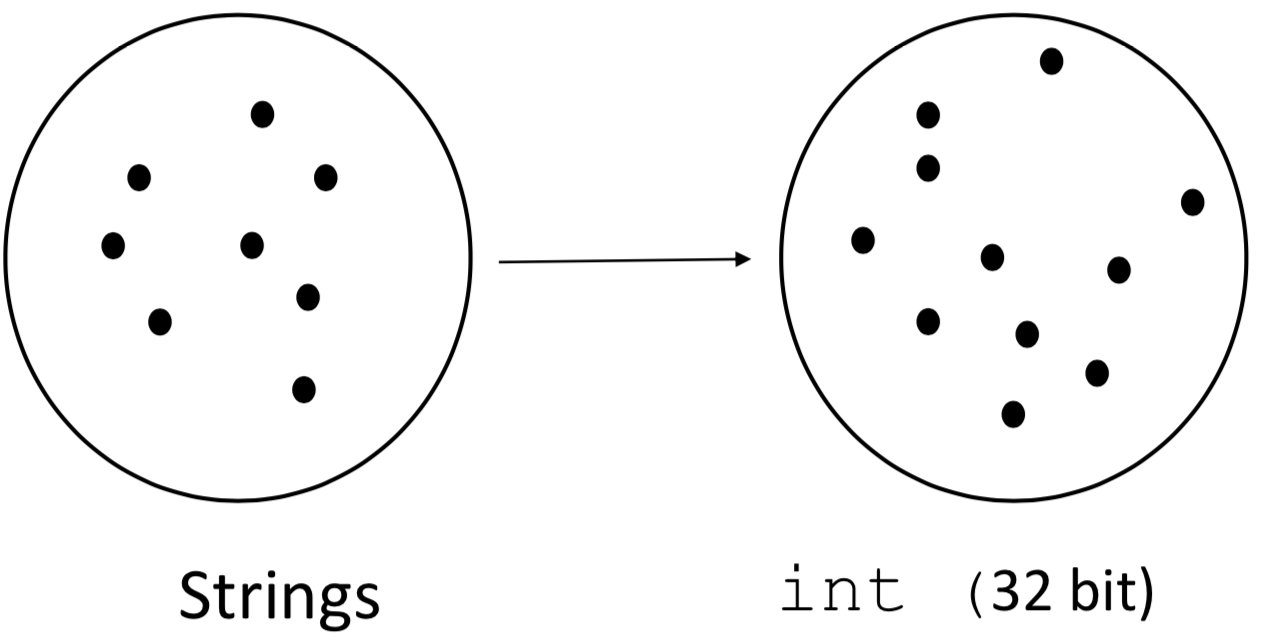
Hash code

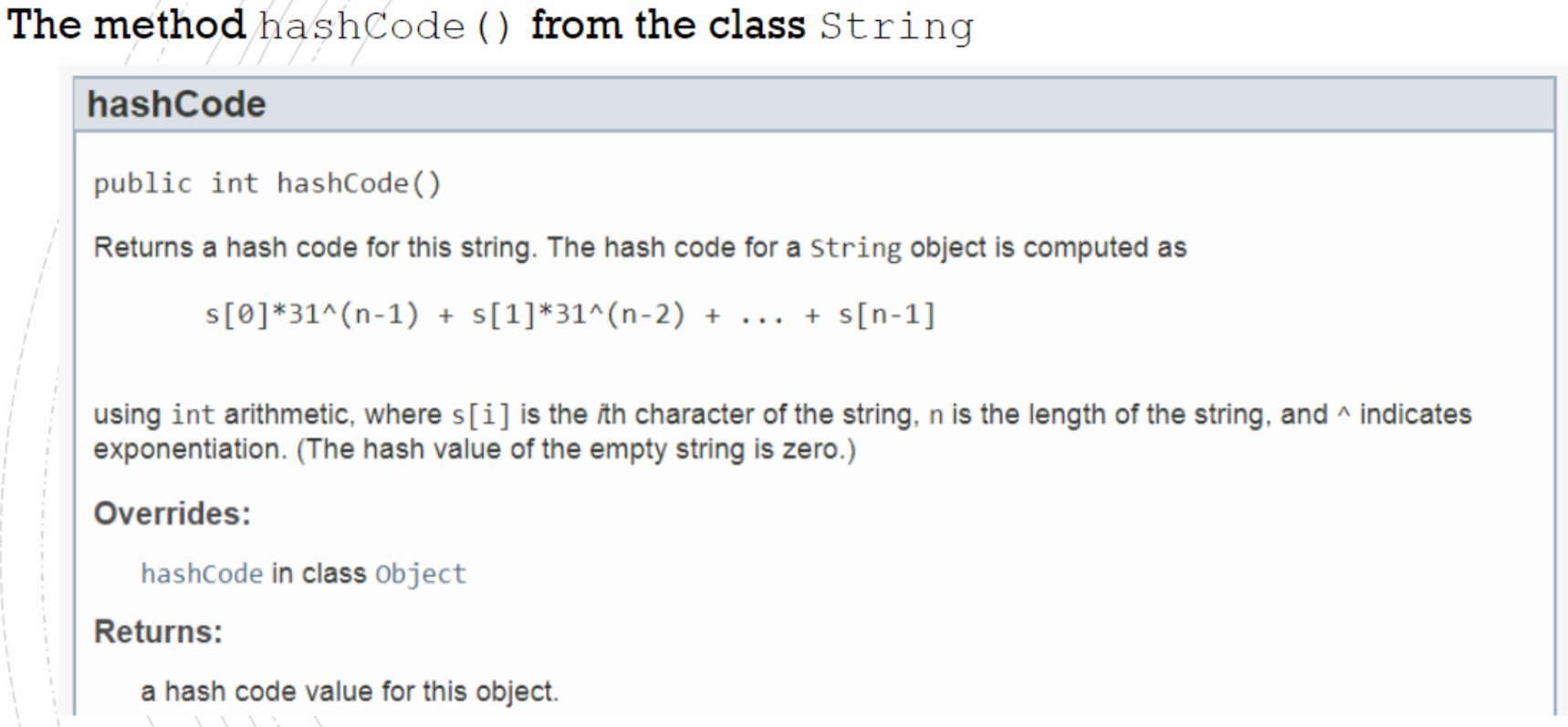
Define a map from keys to large range of positive integers. Such a map is called a hash code.

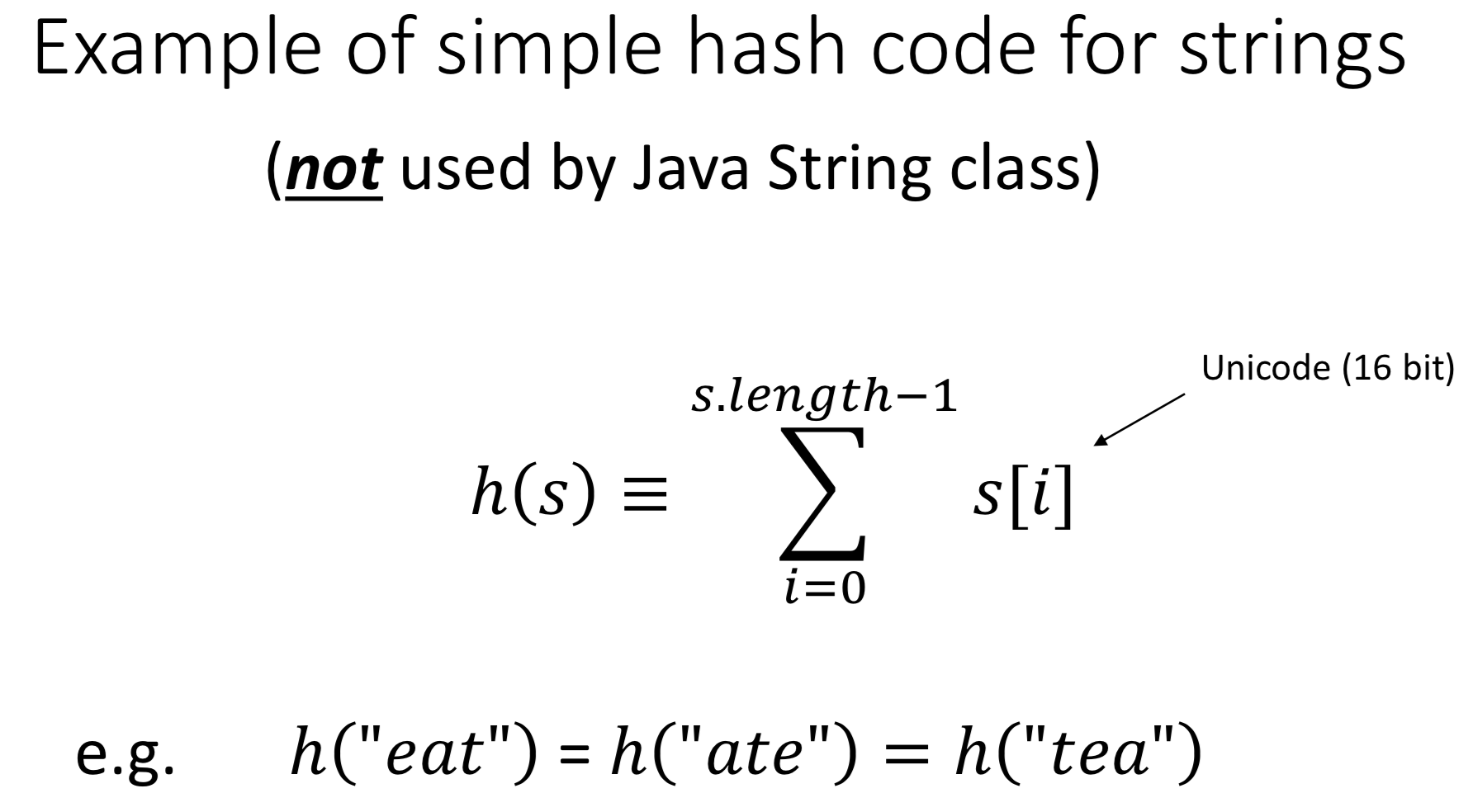
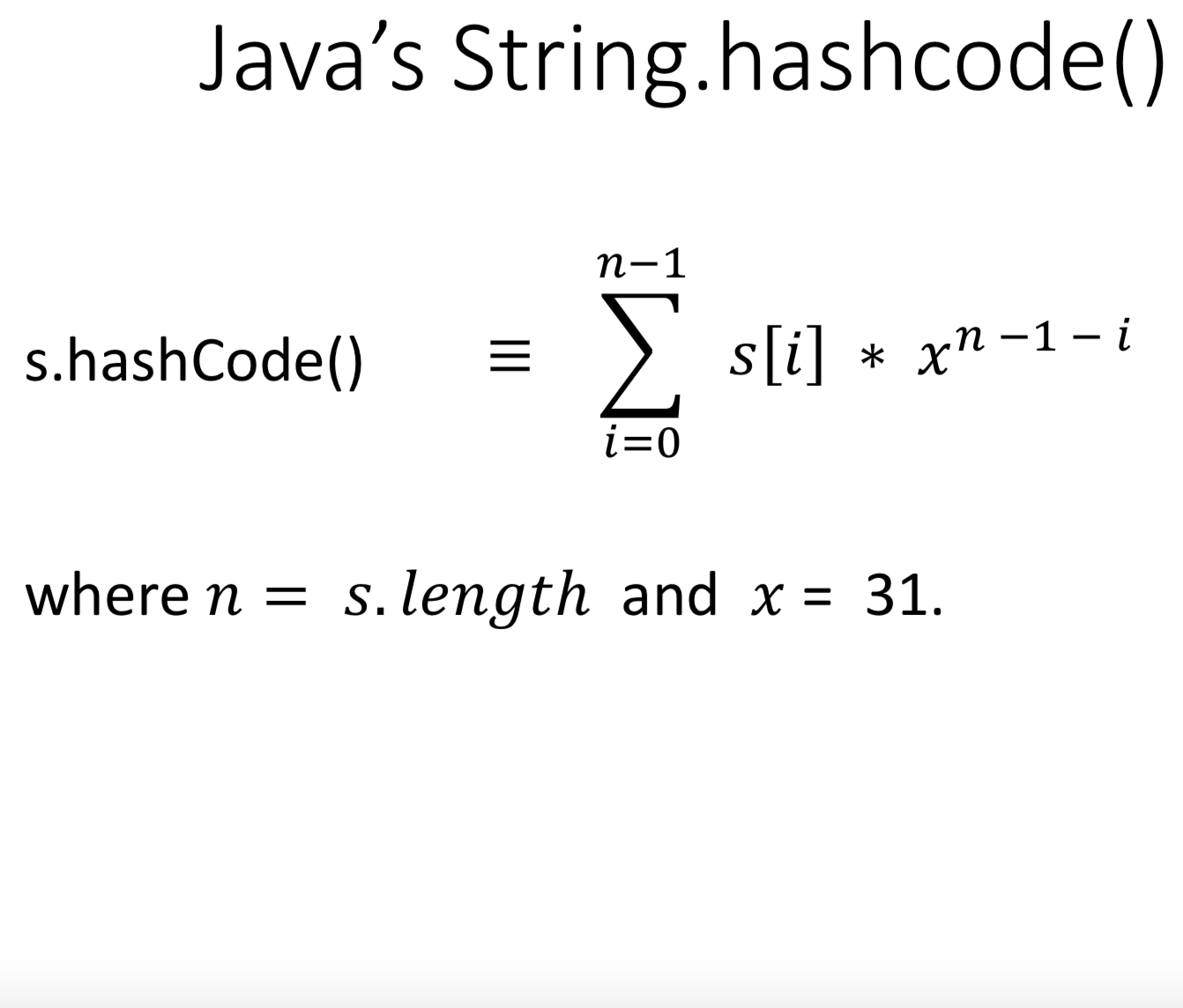


If a class doesn’t override Object.hashCode() then “obj1.hashcode() == obj2.hashcode()” is equivalent to “obj1 == obj2”.

***Java’s String.hashcode()-***For each String, it defines an integer.





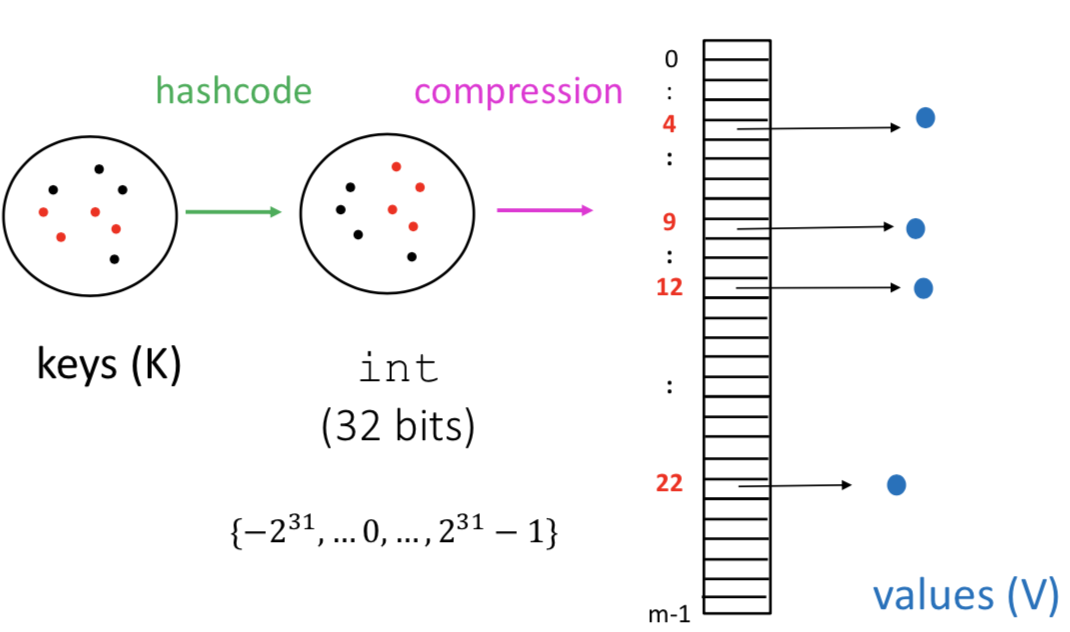
 

However, we still CANNOT conclude that s1.equals(s2) when s1.hashCode() == s2.hashCode().

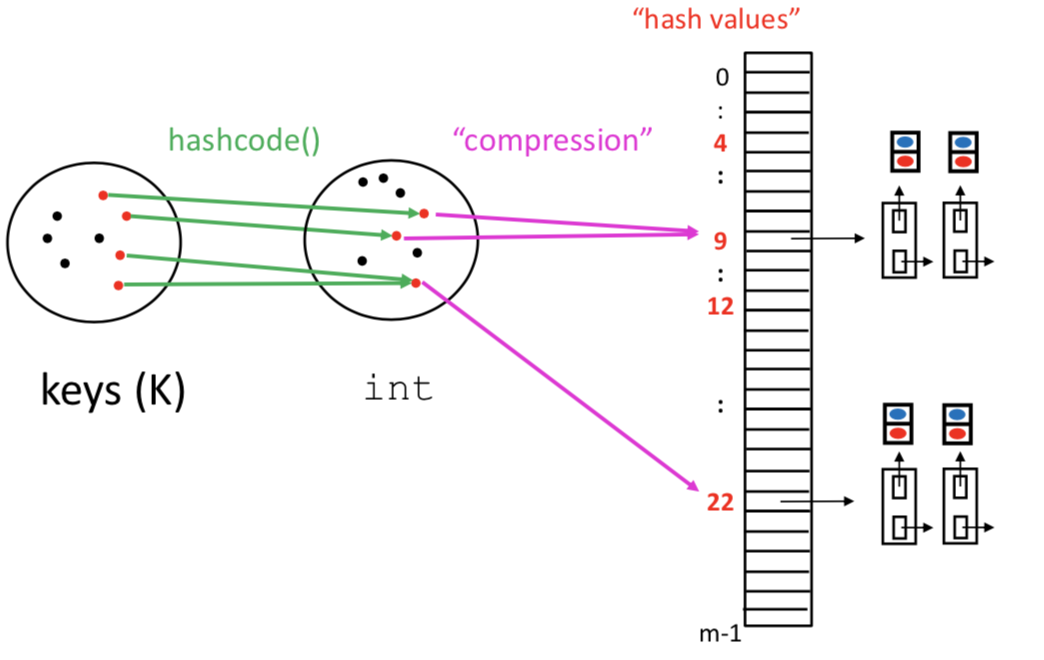
ASIDE: Java uses Horner’s rule for efficient polynomial evaluation(not testable)

Hash map & hash table

compression map: 𝑖 → | 𝑖 | 𝑚𝑜𝑑 𝑚, where m is the length of the array



“hash function” ≡ compression ° hashCode : key 🡪 hash value {0, ... , m-1}

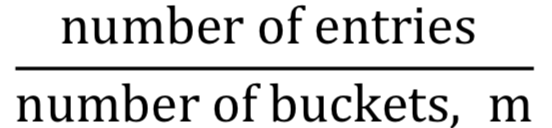


**Collision:**

when two or more keys k map to the same hash value. (two ways)

Each array slot + linked list is called a **bucket**.

So there are m buckets.

**Load factor of hash table**= 

One typically keeps the load factor below 1.

In the Java HashTable and HashMap classes, the default load factor is 0.75.

[Performance of Hash Maps]

• put(key, value)

*If load factor is less than 1 and if hash function is good, then these operations are O(1) “in practice”.*

• get(key)

• remove(key)

*It will need to look at each bucket and search its linkedlist for that value. These last three methods all require traversing the hash table, which takes time 𝑂(𝑛 + 𝑚) where 𝑛 is number of entries. ---max(n,m)*

• contains(value)

• getKeys()

• getValues()

[Java HashMap <K, V> class]

In constructor, you can specify initial number of buckets, and maximum load factor

• How is hash function specified ?

Use key’s hashCode(), take absolute value, and compress it by taking mod of the number of buckets.

Java HashSet<E> class

Similar to HashMap, but there are no values. store a set of objects of some type.

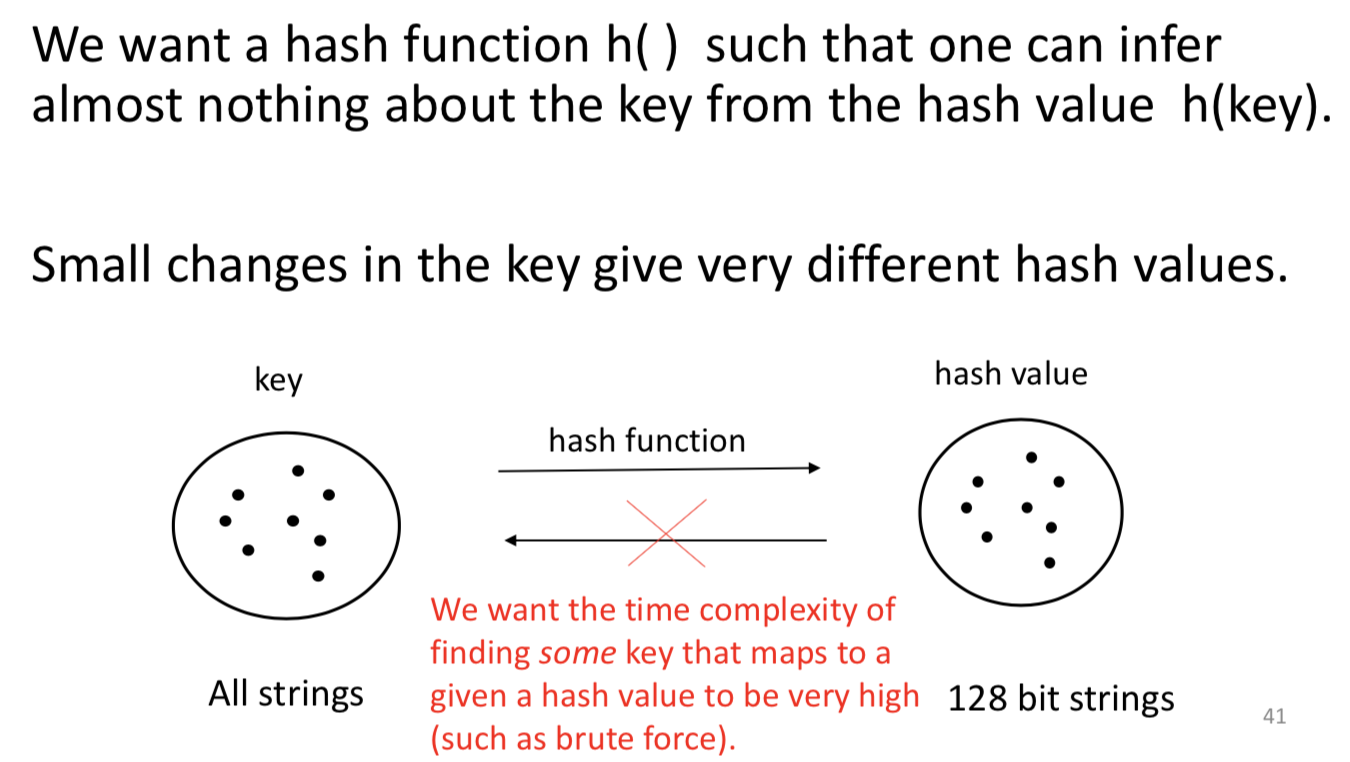
If hash function is good, then these operations are O(1).

Note that this is not a list! There is no 1st, 2nd, .... element.

<Password Authentication(secure)>

The map {(username, h(password))} is stored in a file on the web server.

* What would the user do? Enter a username and password.
* What would the web server do ? Hash the password and compare to entry in map, and throw away the passord.
* What could a mischievous hacker try to do? “Brute force” or “dictionary” attack.



Do not confuse hashing with encryption/decryption.

