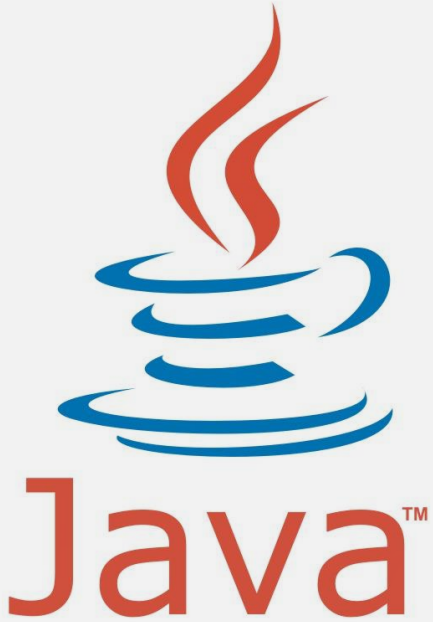
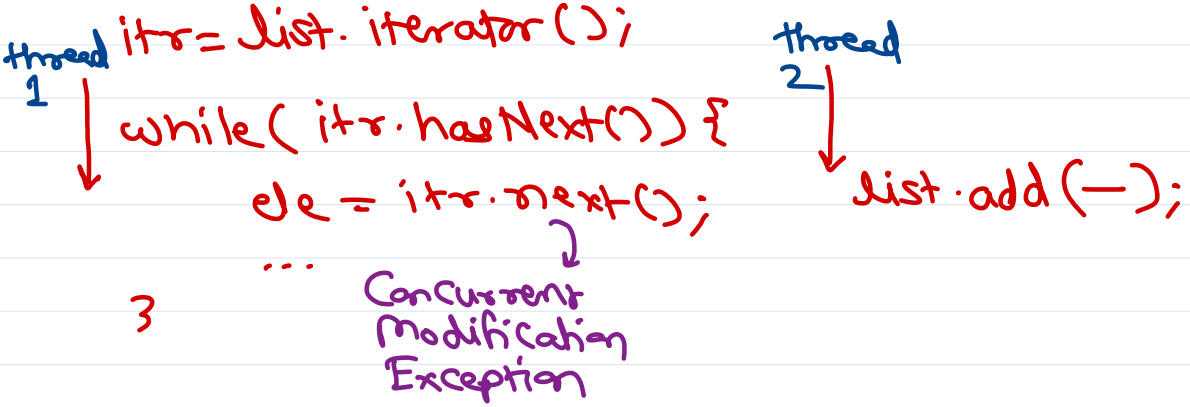


Core Java

Trainer: Nilesh Ghule

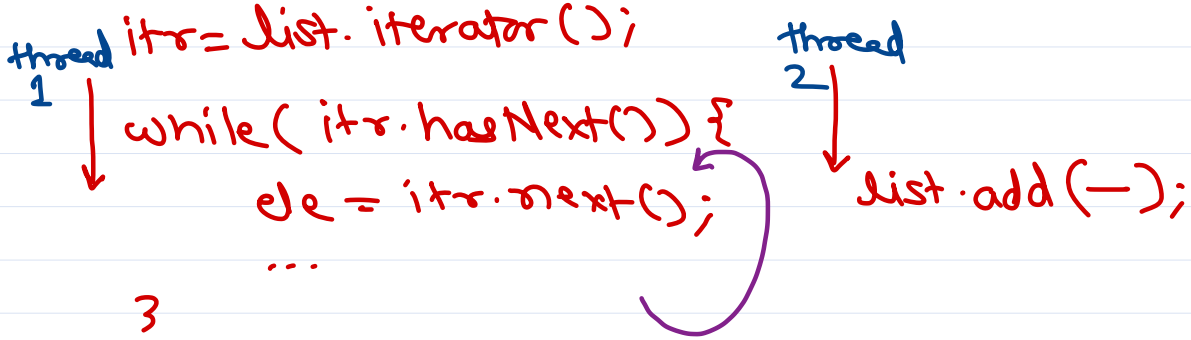
Iterators

fail-fast iterator



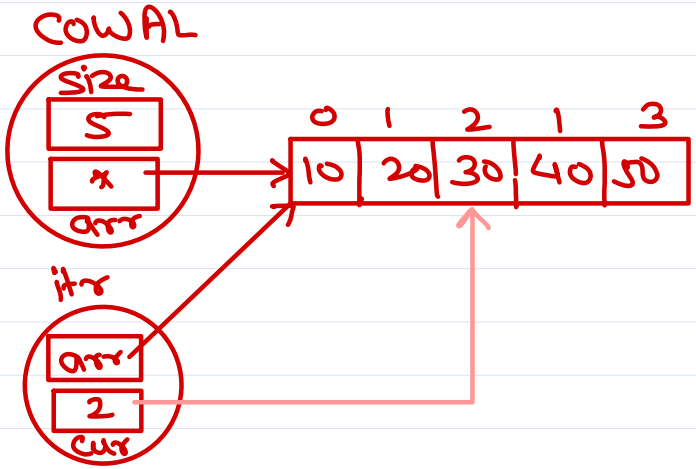
Collections in java.util pkg have fail fast iterators.

fail-safe iterator

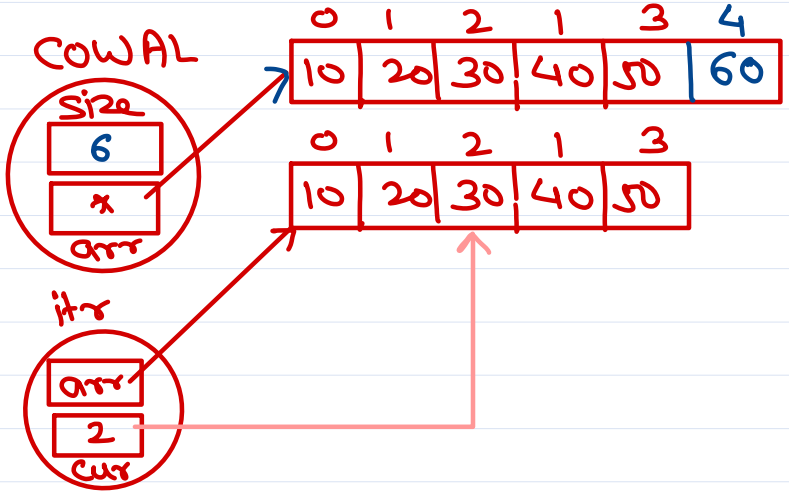


Collections in java.util.concurrent pkg have fail-safe iterators.

CopyOnWriteArrayList

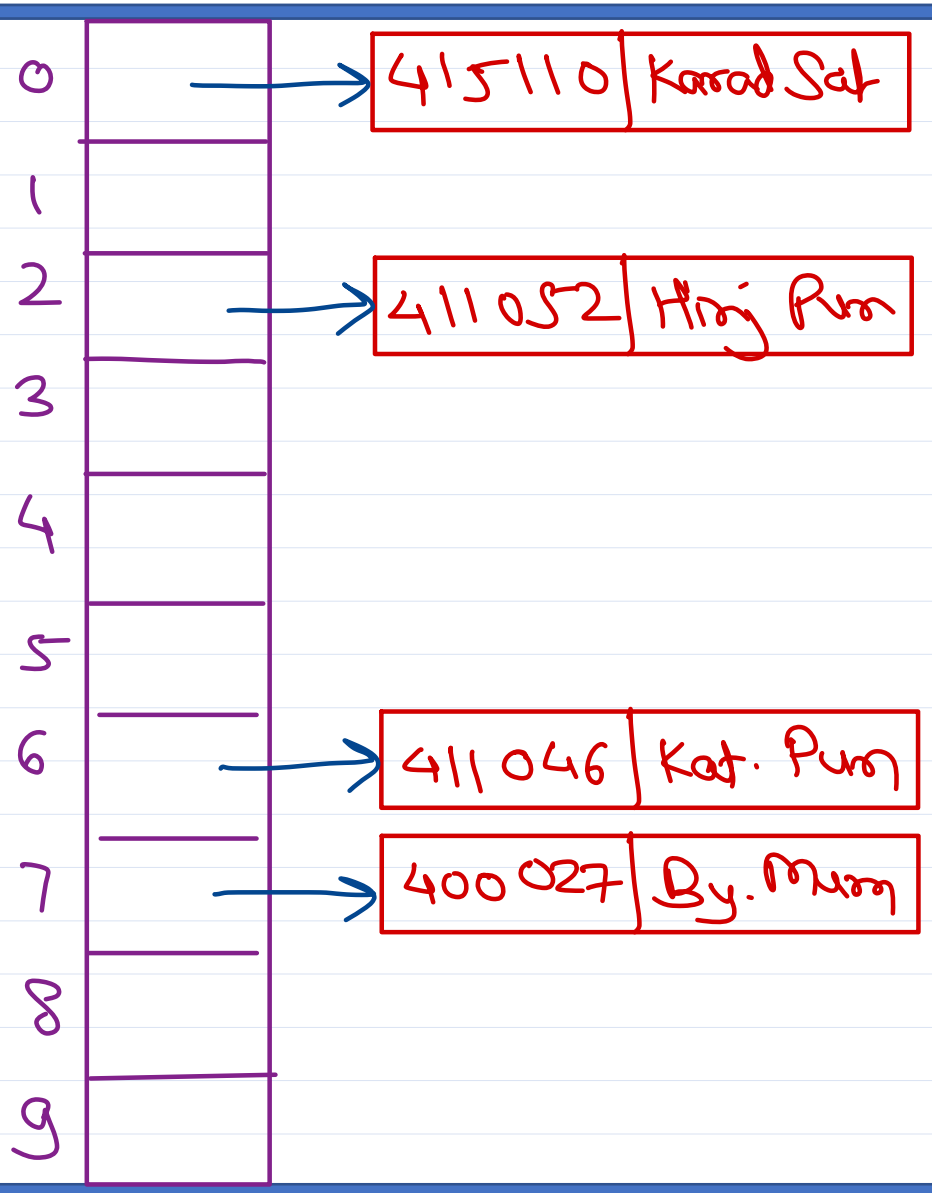



```
itr = list.iterator();
while (itr.hasNext()) {
    ele = itr.next();
    ...
}
```




```
list.add(60);
```

Hash table in Data structure



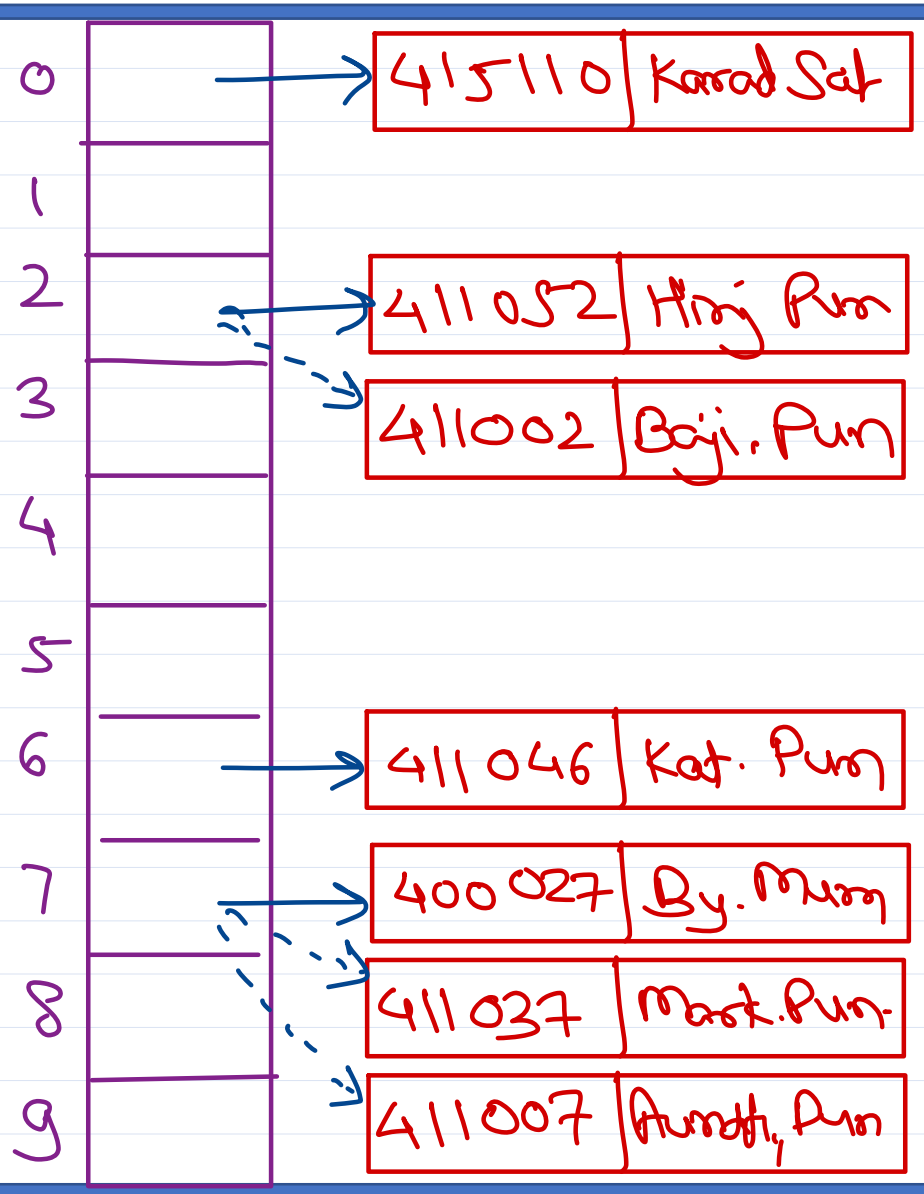
Hash table is a DS in which ele stored in key-value pair so that for a given key, value can be searched in fastest possible time. Ideal time complexity : O(1).

e.g. Contacts : name → mobile.
regions : pin → region.
Stud mgmt: Student → progress card

$h(k) = k \% 10$

hash fn is math fn of key, that decides the slot of the table in which ele is to be stored/searched.

Hash table in Data structure



$h(k) = k \% 10$

if different keys yields same slot in table (by hash fn) then collision occurs.

collision handling techniques

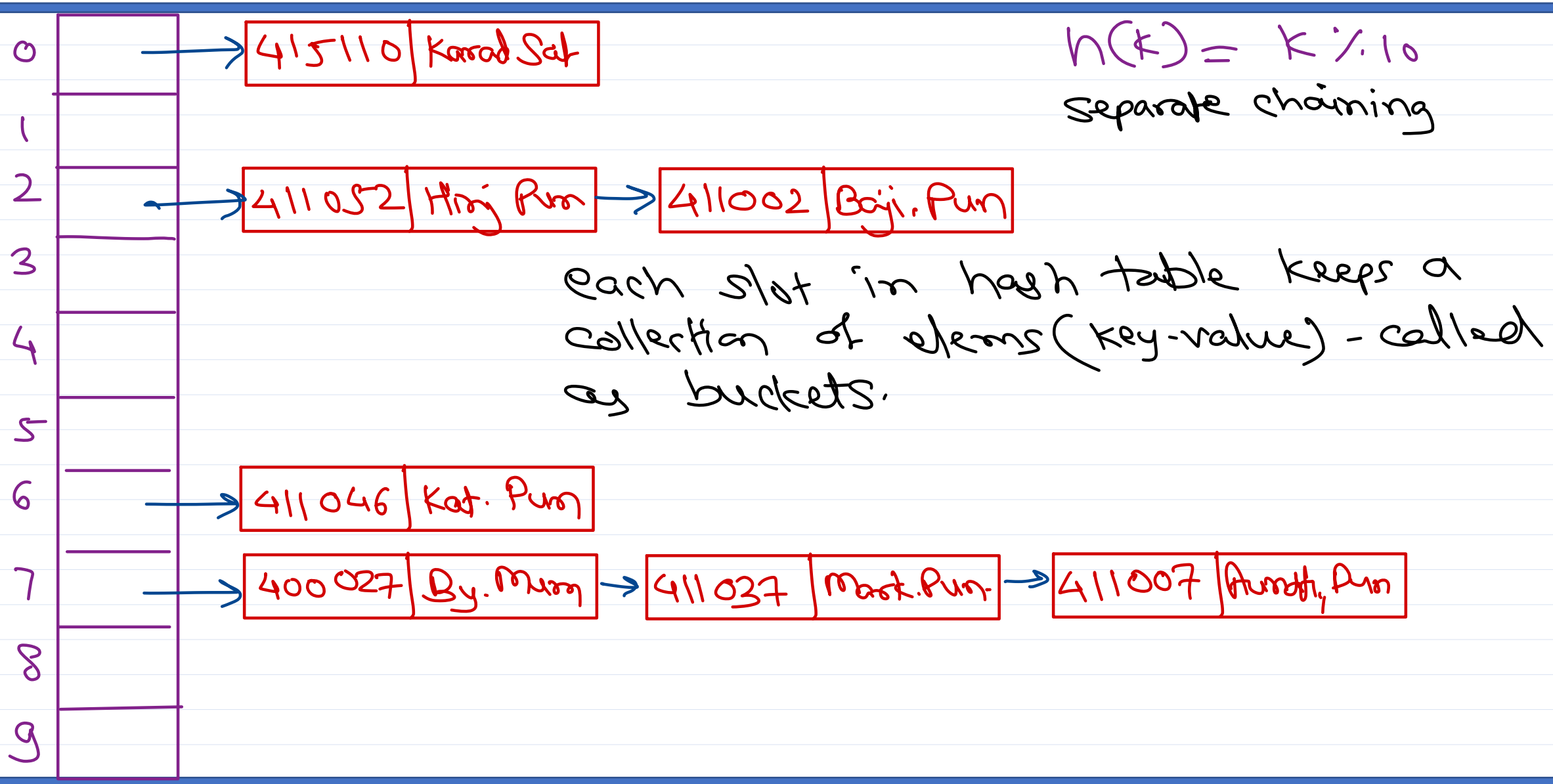
1. open addressing
2. separate chaining.

load factor = num of entries ÷ num of slots

e.g.

num of entries	num of slots	load factor
7	10	0.7
10	10	1.0
15	10	1.5

Hash table in Data structure



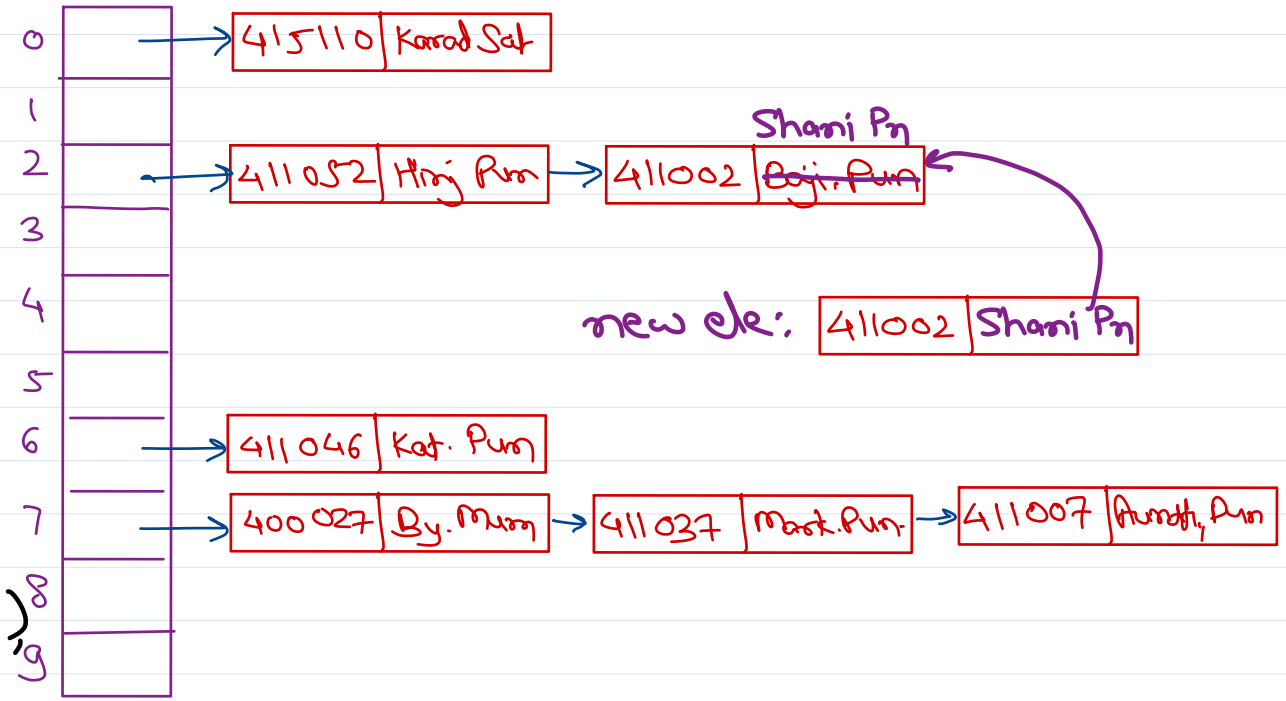
Hash tables in Java

1. HashMap<K,V>
2. LinkedHashMap<K,V>
3. TreeMap<K,V>
4. Hashtable<K,V>

Duplicate keys are not allowed.

If duplicate key is added, new value will replace old value.

```
map.put(key, value);
value = map.get(key);
```



In Java, hash fn is implemented by overriding hashCode() method.

```
slot = key.hashCode() % num of slots;
```

Ideal hashCode() impl should return diff hash code for diff objects.

if two objs are not equal (by equals()), then their hash code should be diff - for better performance.

if hash codes are same for diff keys, searching in hash table will be slow.

if two objs are equal (by equals()), then their hash code must be same.

Sets vs Maps

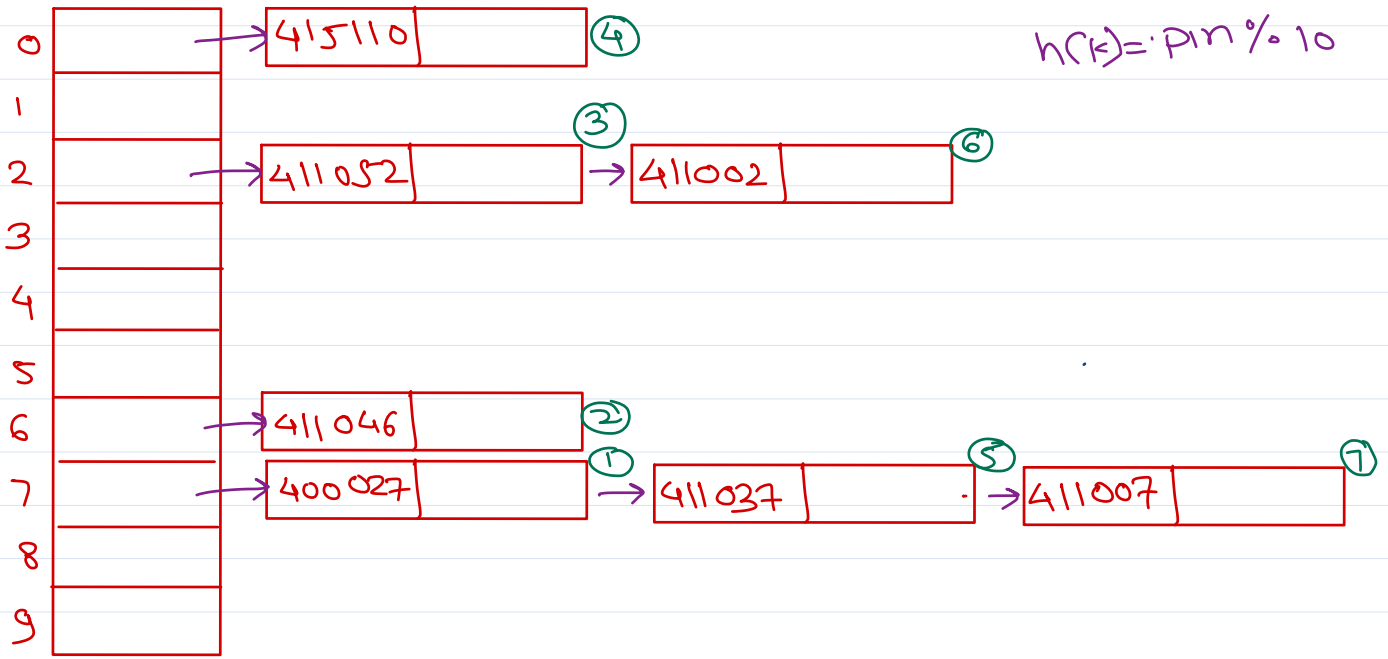
HashSet<K> = HashMap<K, null>

LinkedHashSet<K> = LinkedHashMap<K, null>

TreeSet<K> = TreeMap<K, null>

```
Set<Integer> s = new HashSet<>();
```

```
s.add(400027);
s.add(411046);
s.add(411052);
s.add(415110);
  :
```



Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>