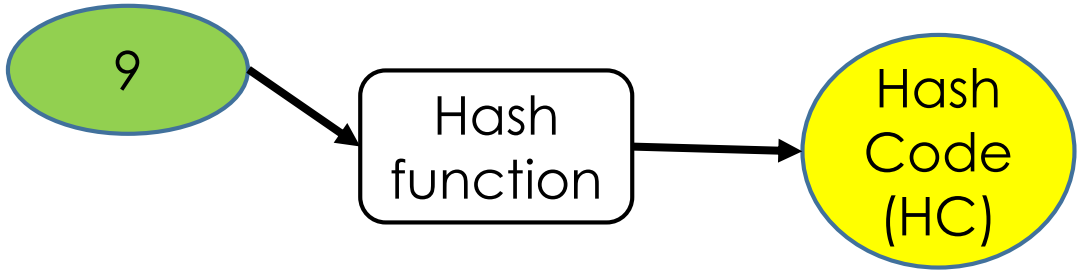
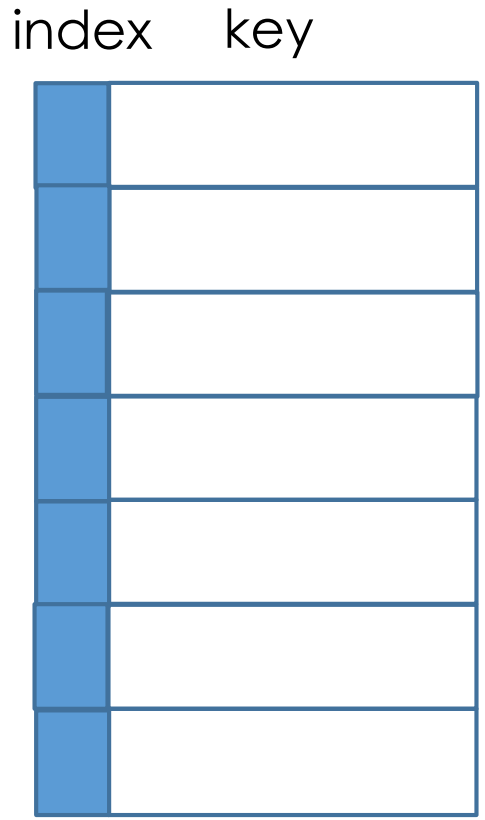
Hash Tables: Collisions

By the end of this video you will be able to…

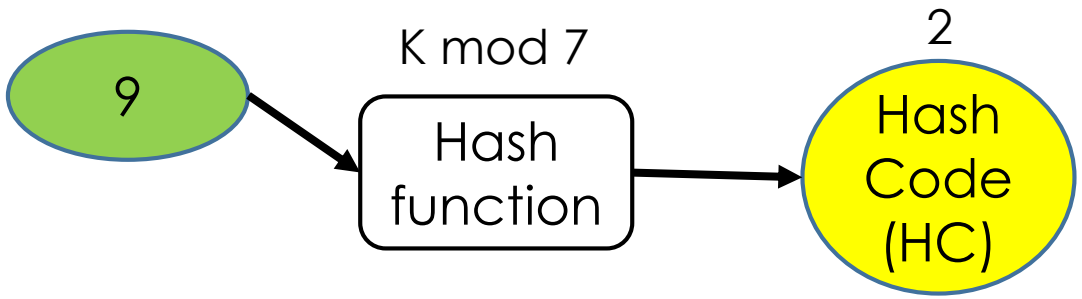
- Describe alternative methods for handling collisions in a Hash Table
- Identify other challenges associated with Hash Tables

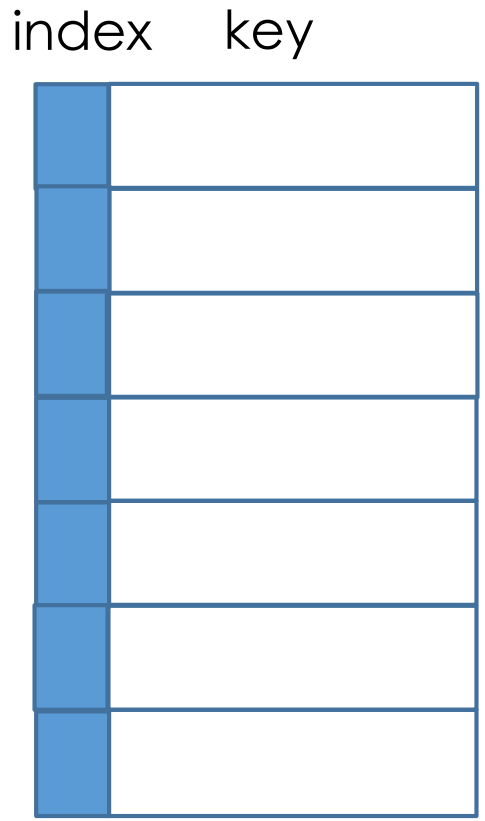
Hash Table Insert



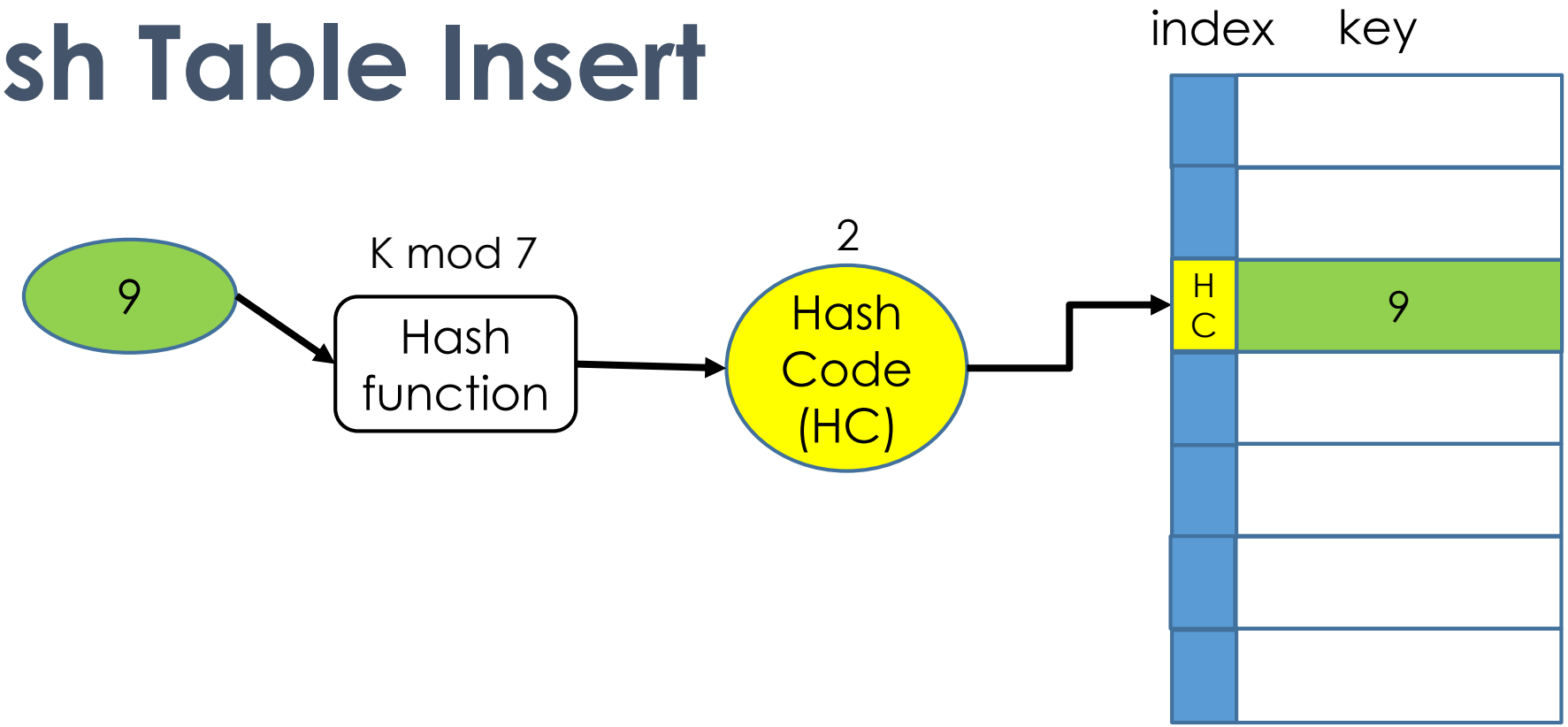


Hash Table Insert

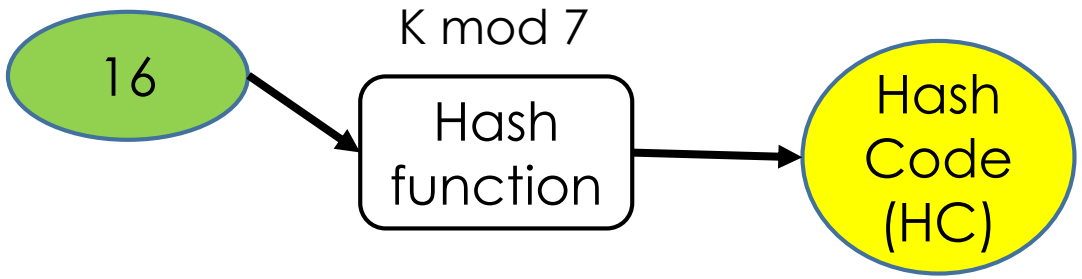


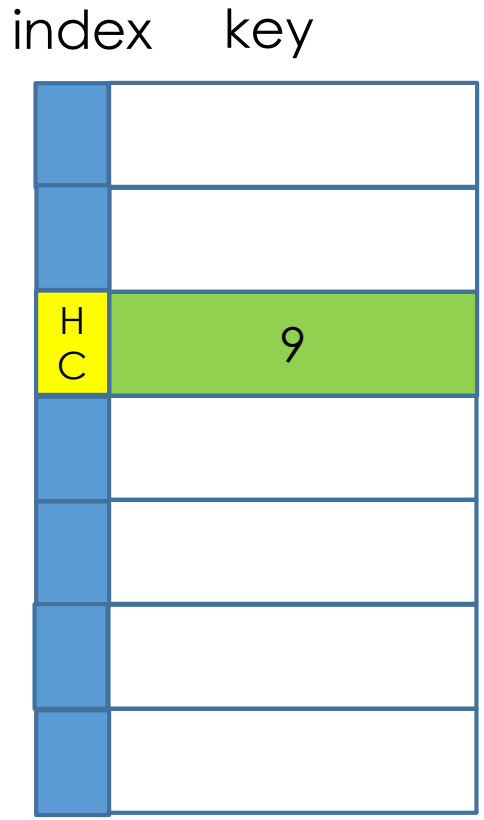


Hash Table Insert

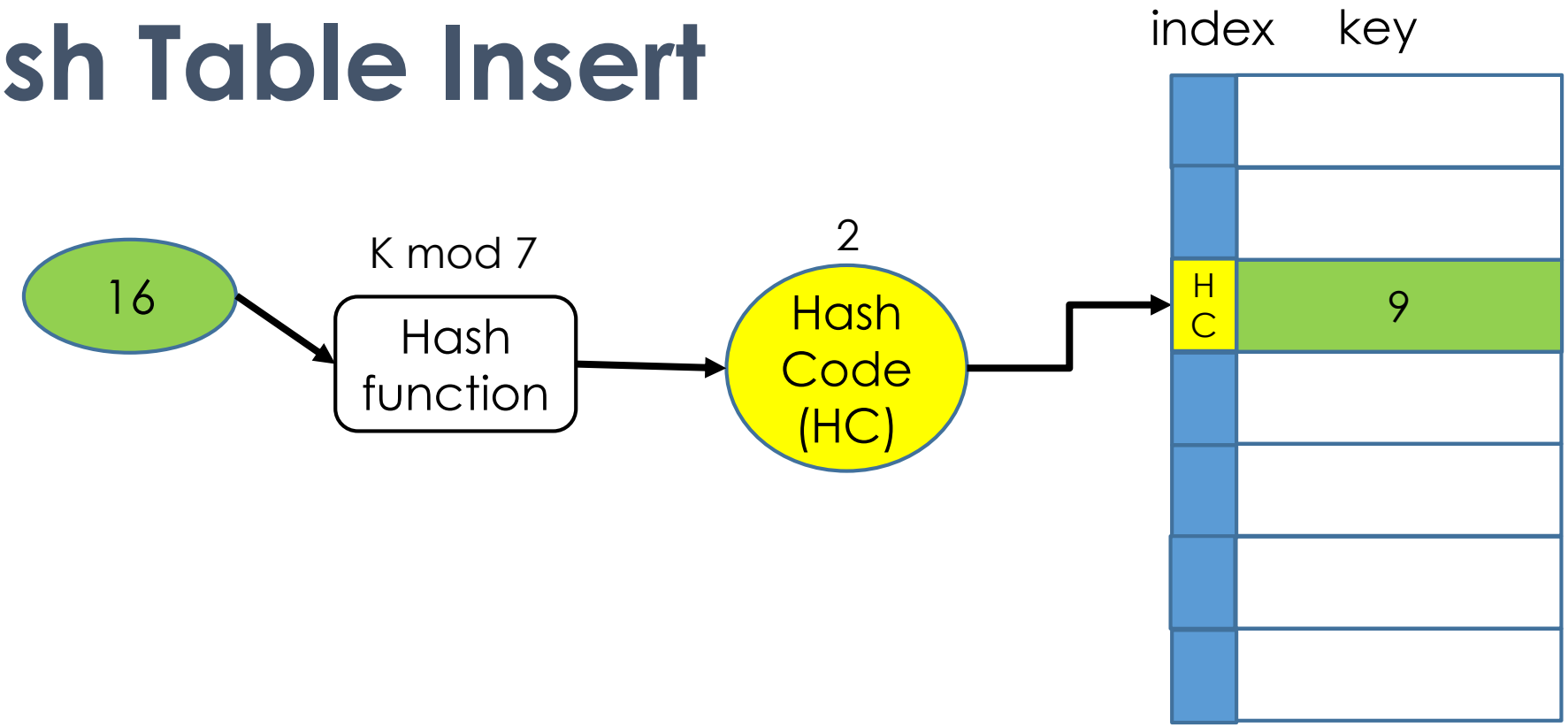


Hash Table Insert

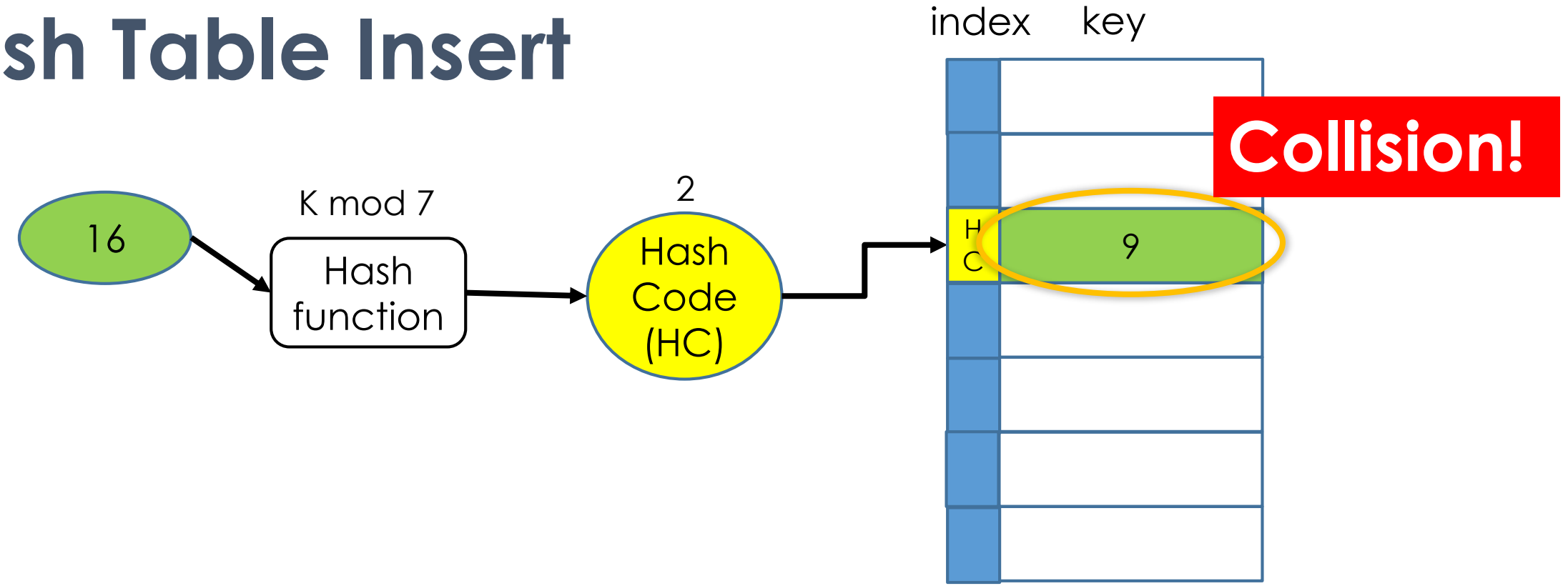




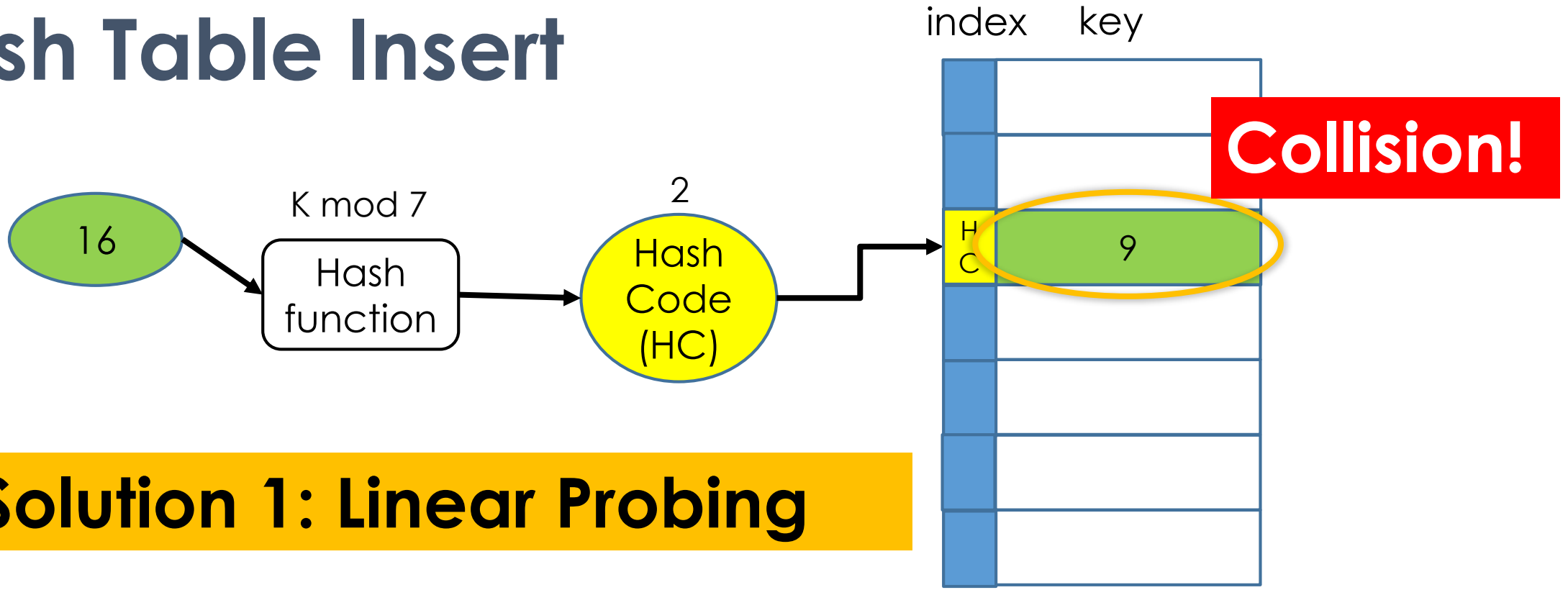
Hash Table Insert



Hash Table Insert

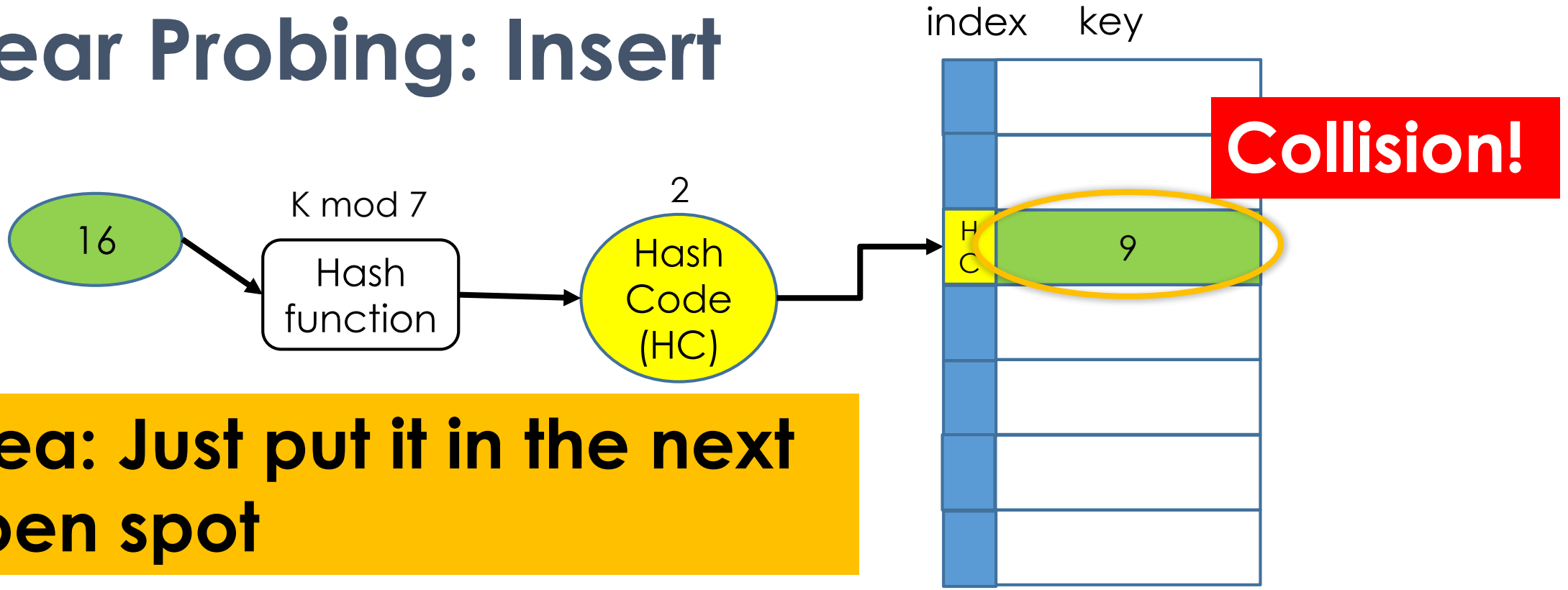


Hash Table Insert



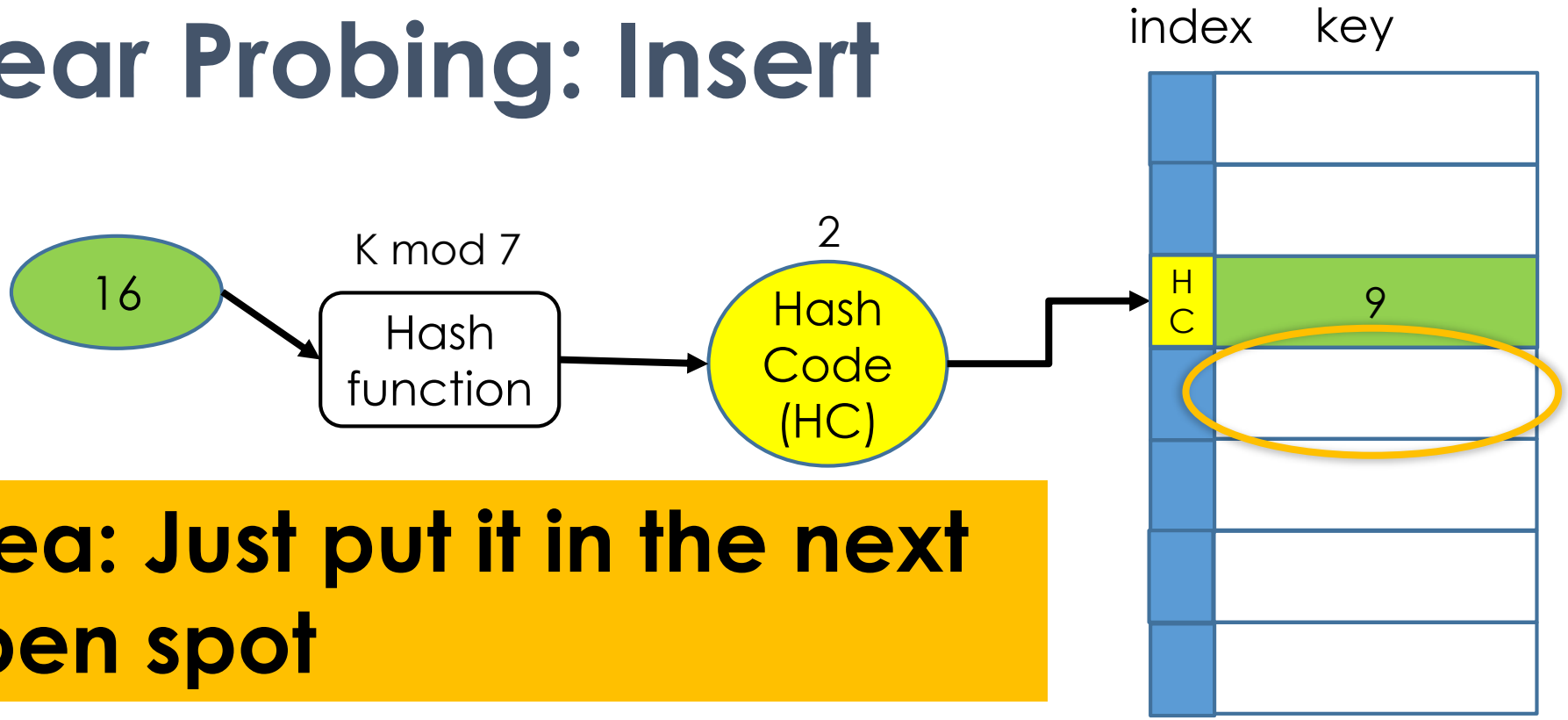
Solution 1: Linear Probing

Linear Probing: Insert



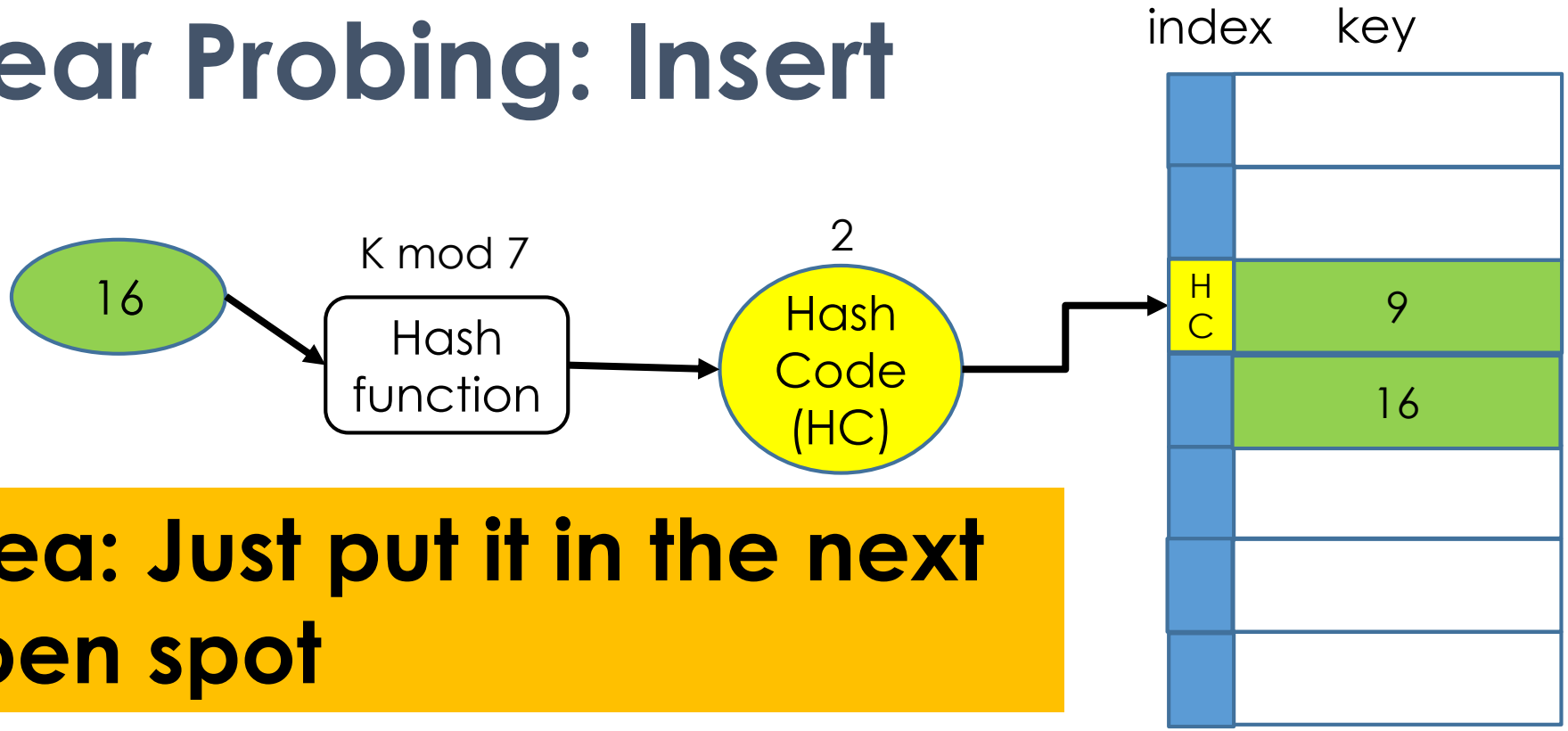
Idea: Just put it in the next open spot

Linear Probing: Insert



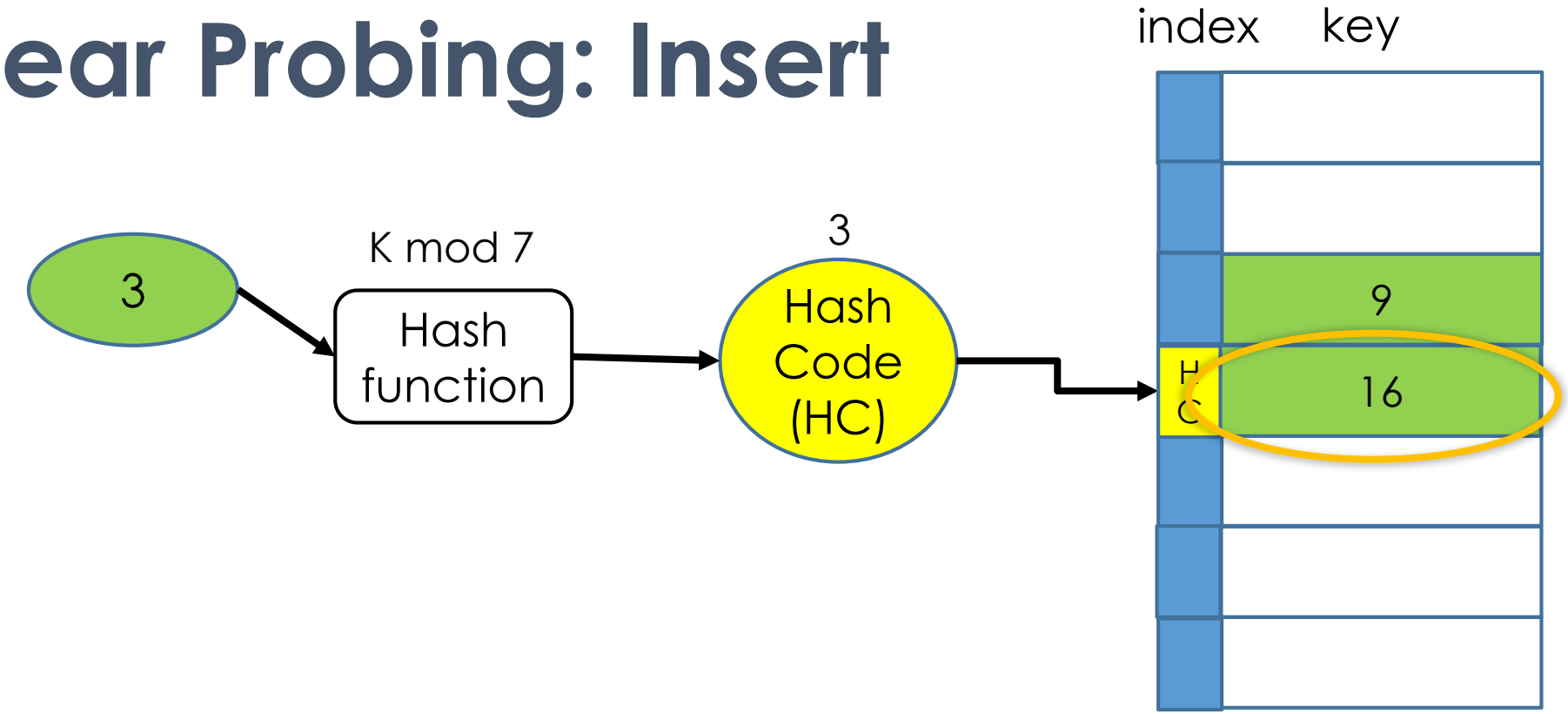
Idea: Just put it in the next open spot

Linear Probing: Insert

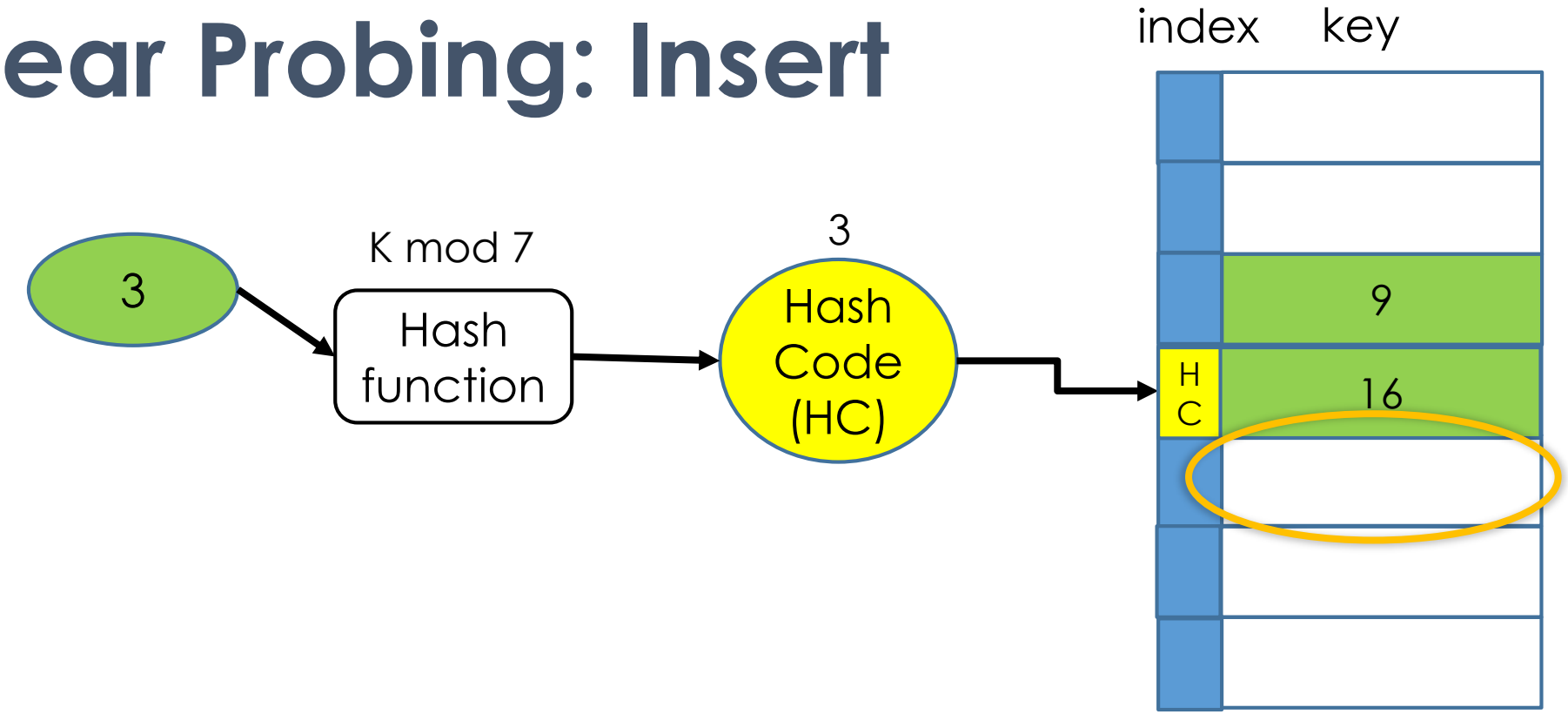


Idea: Just put it in the next open spot

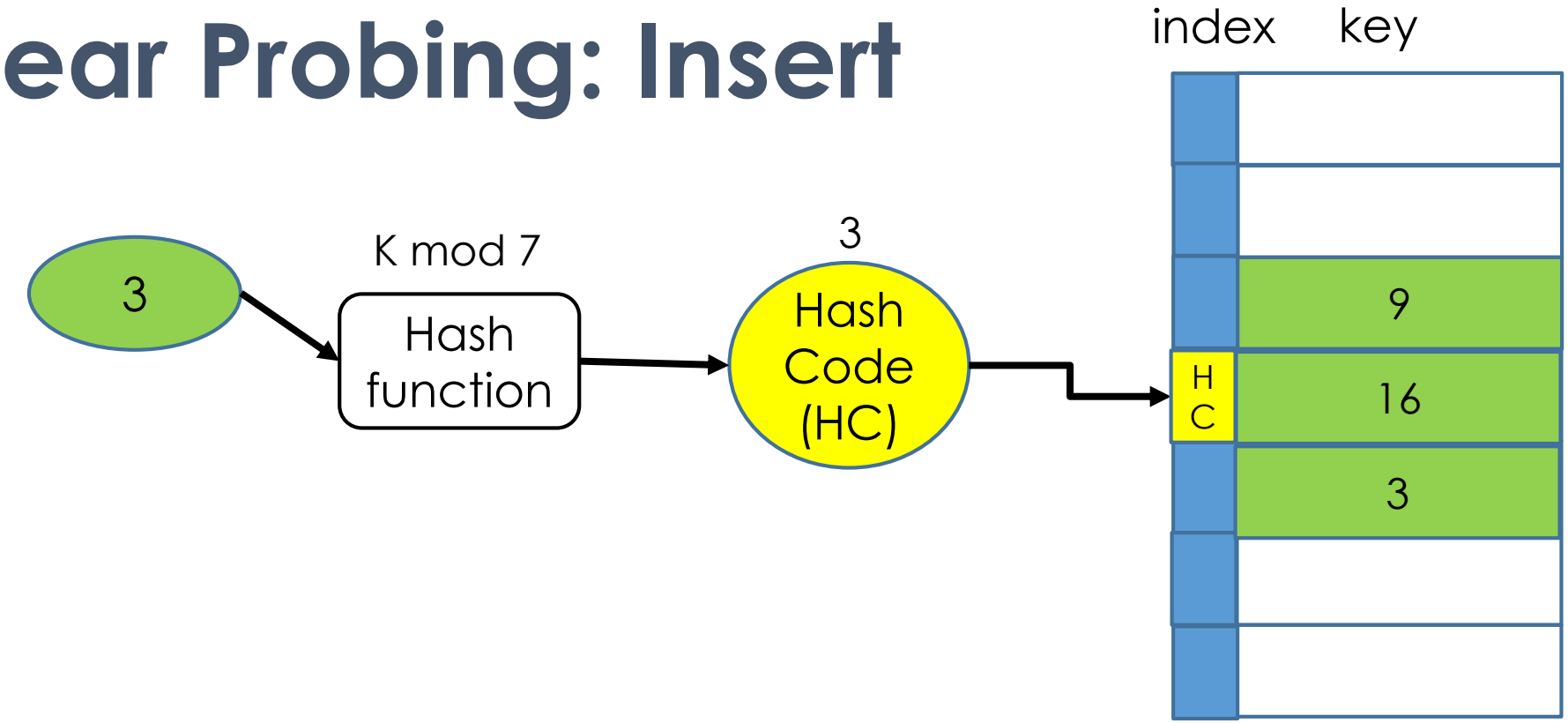
Linear Probing: Insert



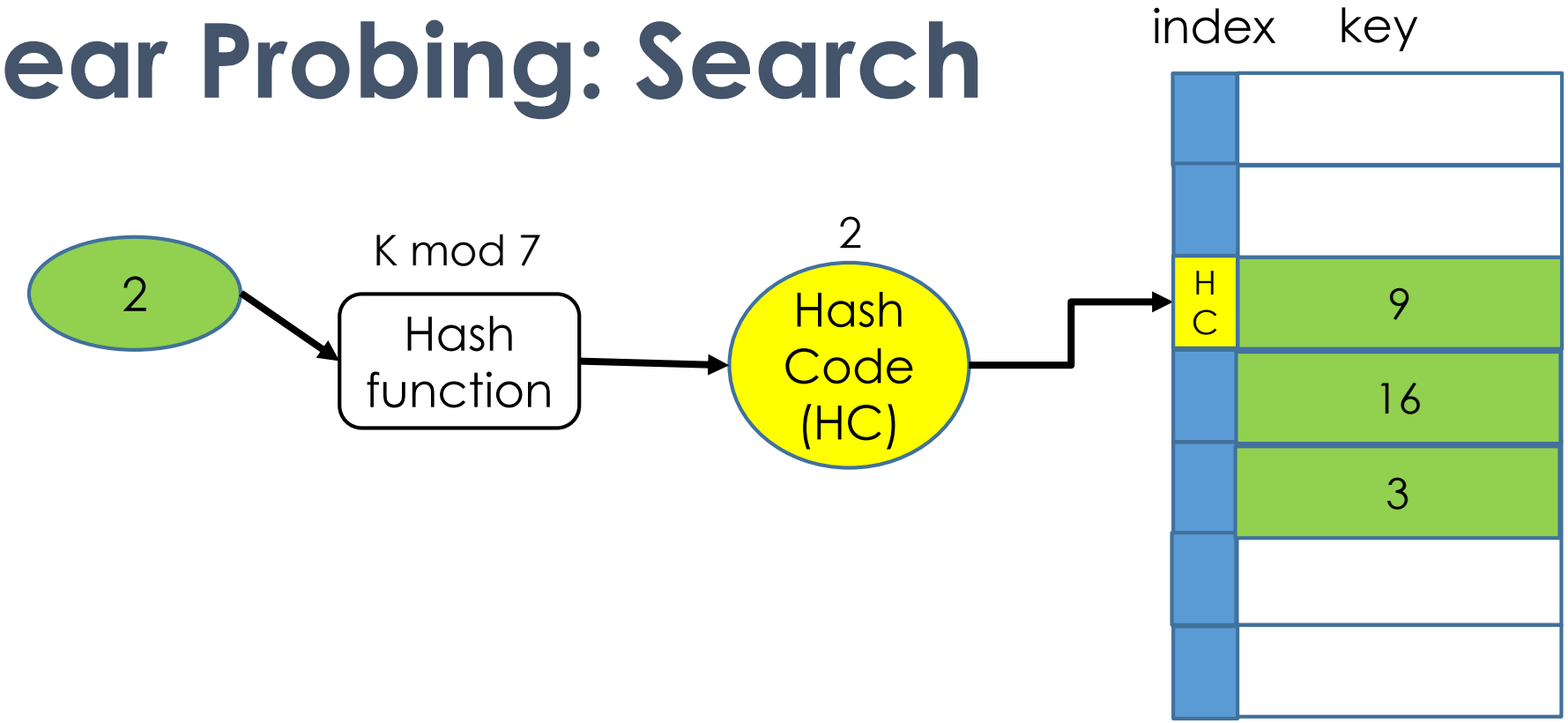
Linear Probing: Insert



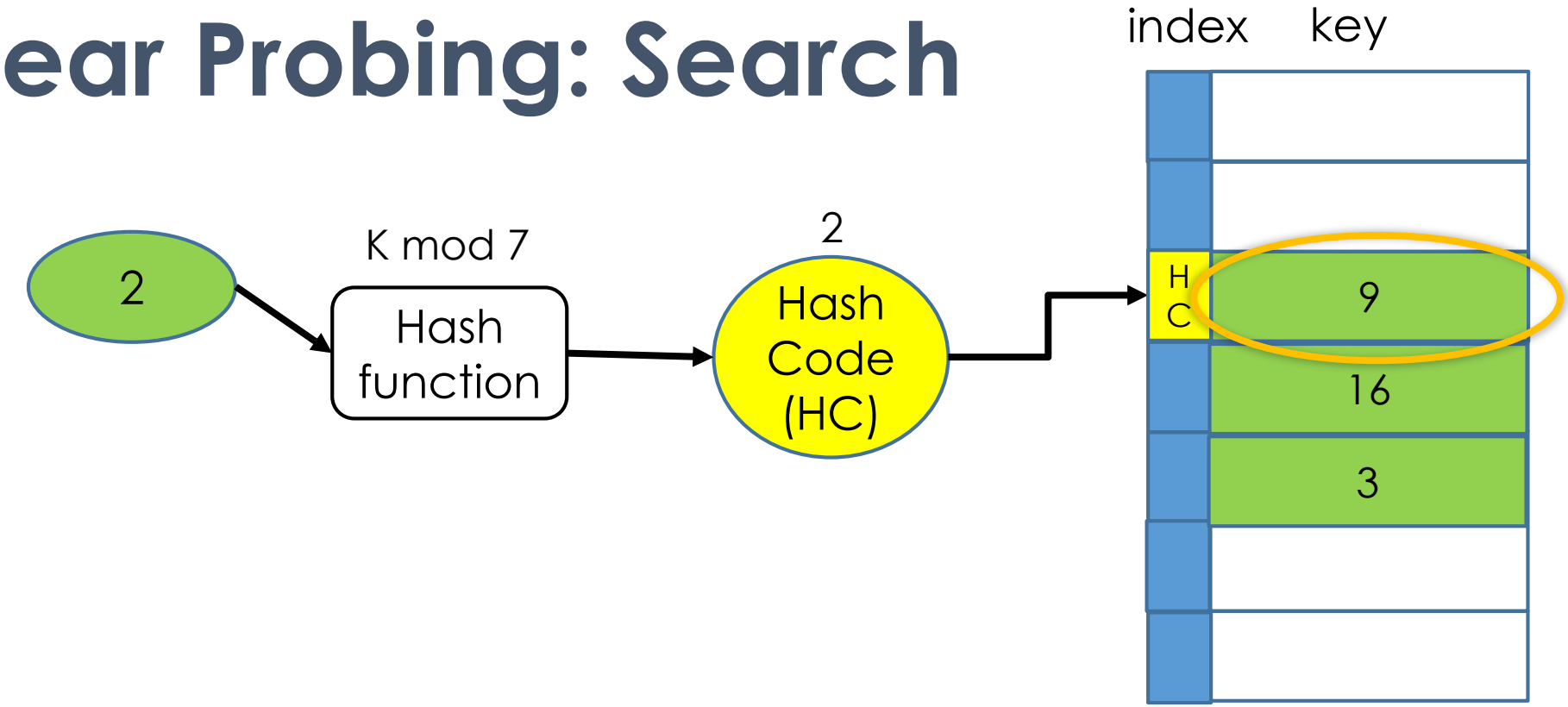
Linear Probing: Insert



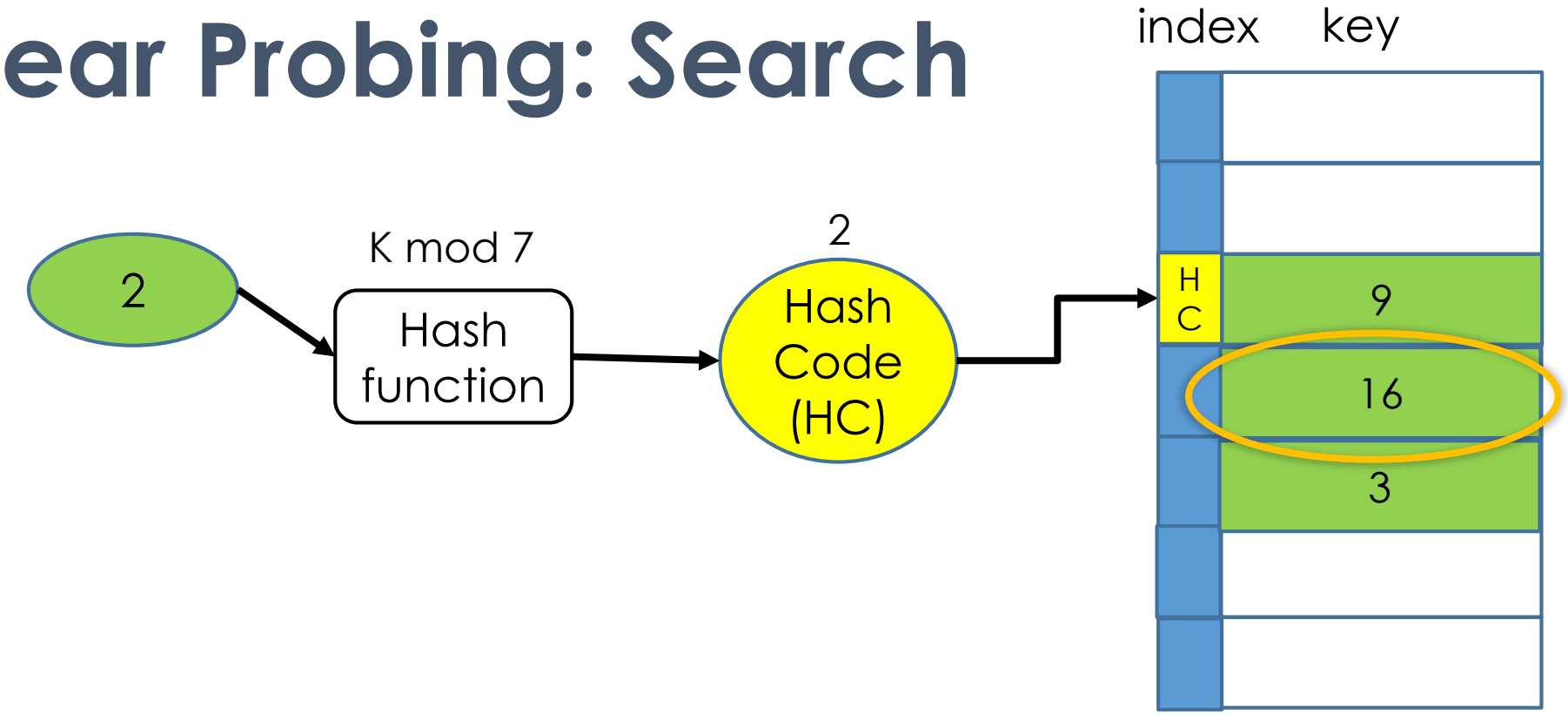
Linear Probing: Search



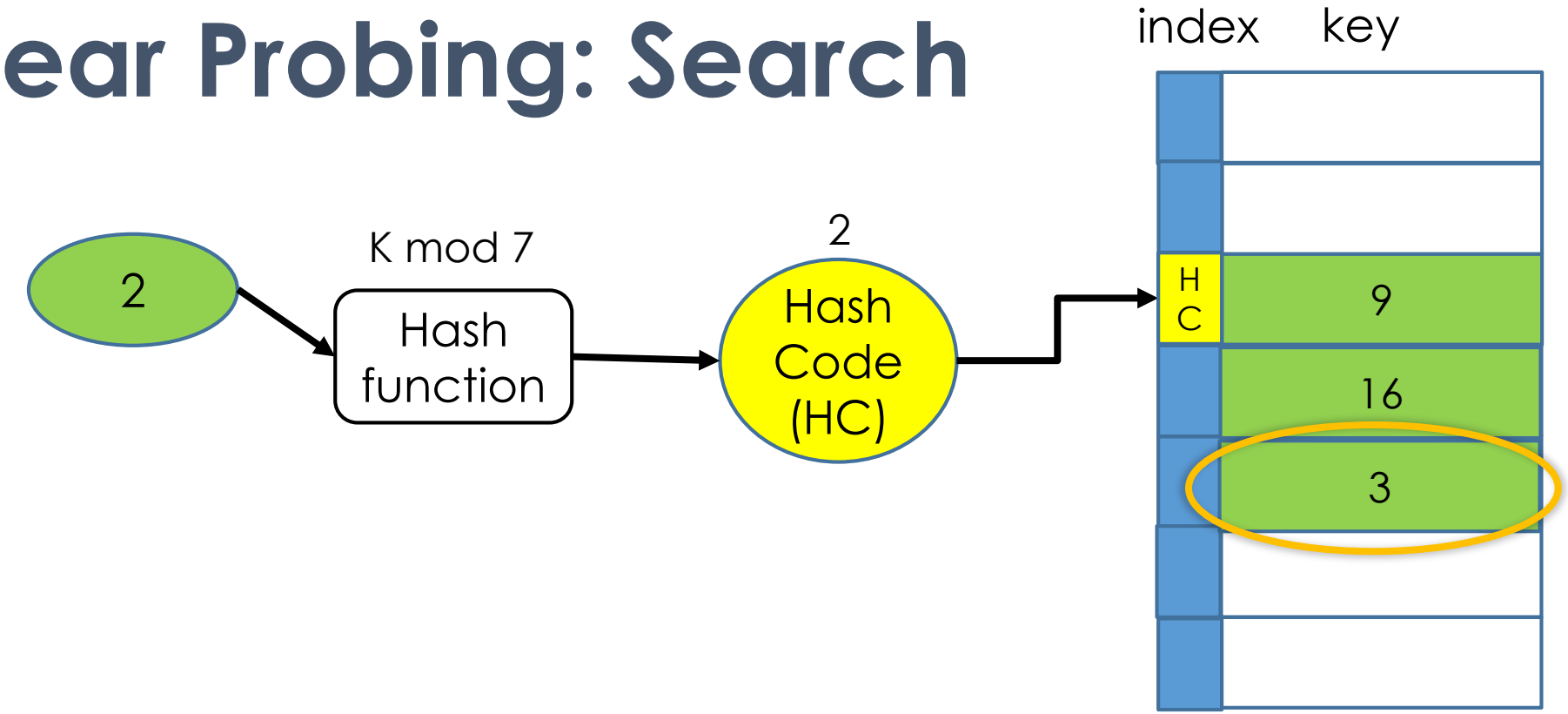
Linear Probing: Search



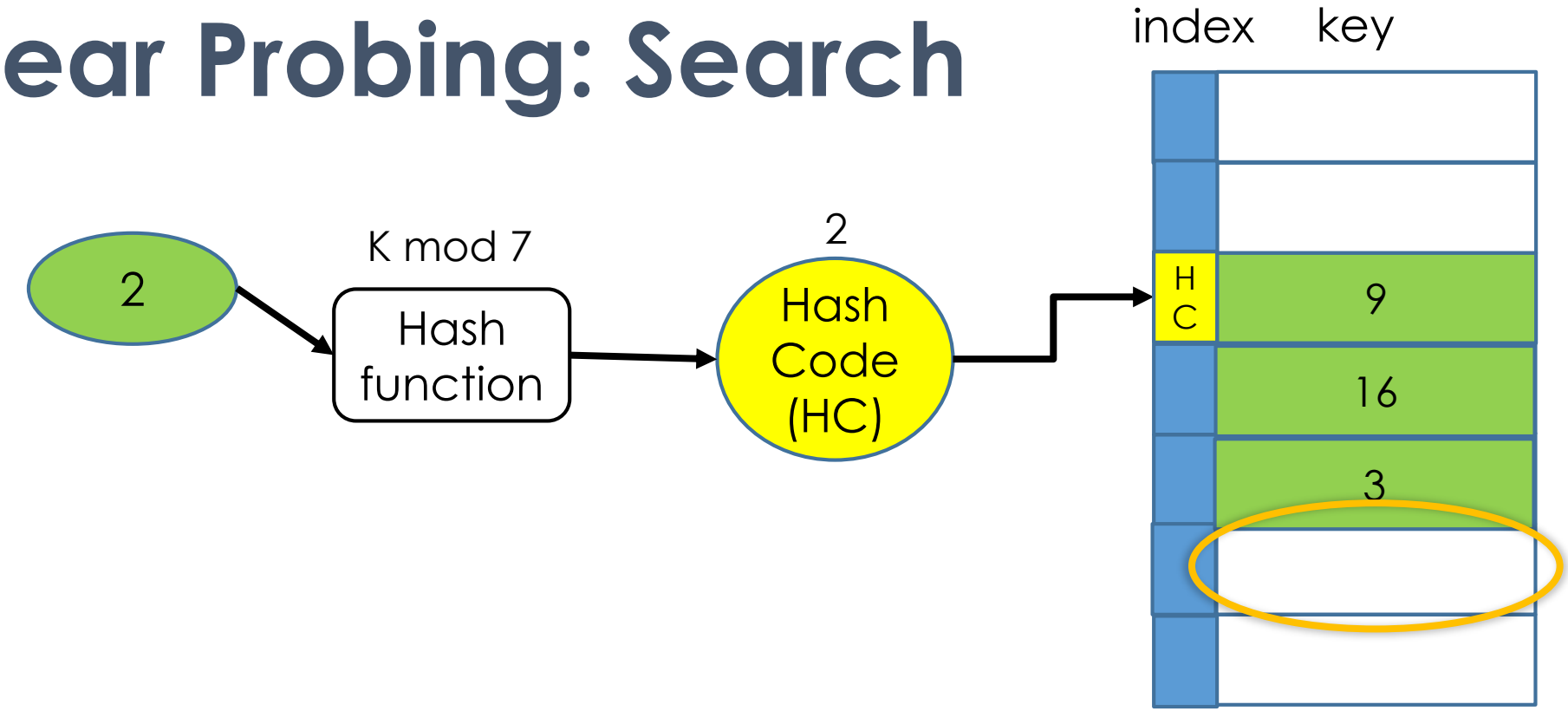
Linear Probing: Search



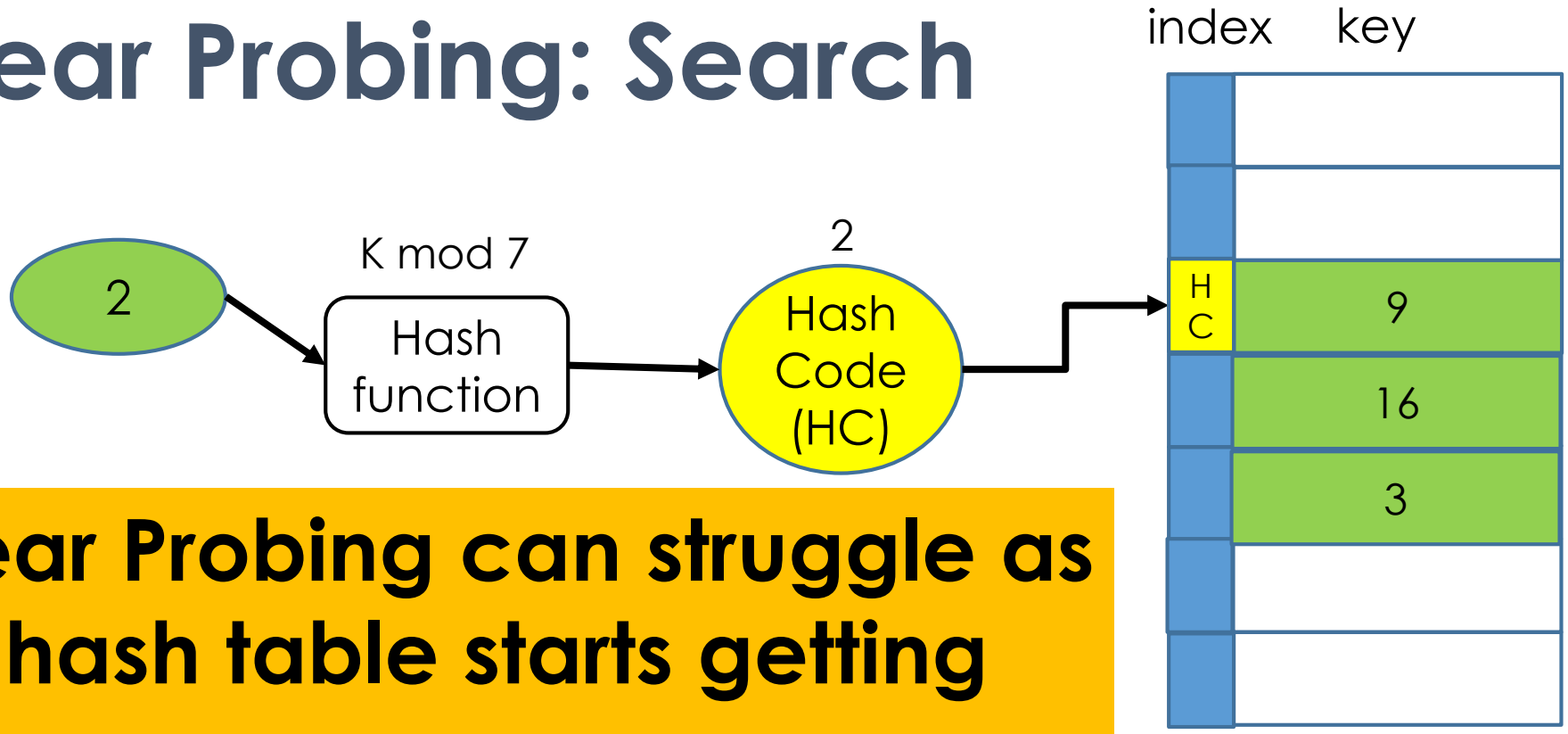
Linear Probing: Search



Linear Probing: Search

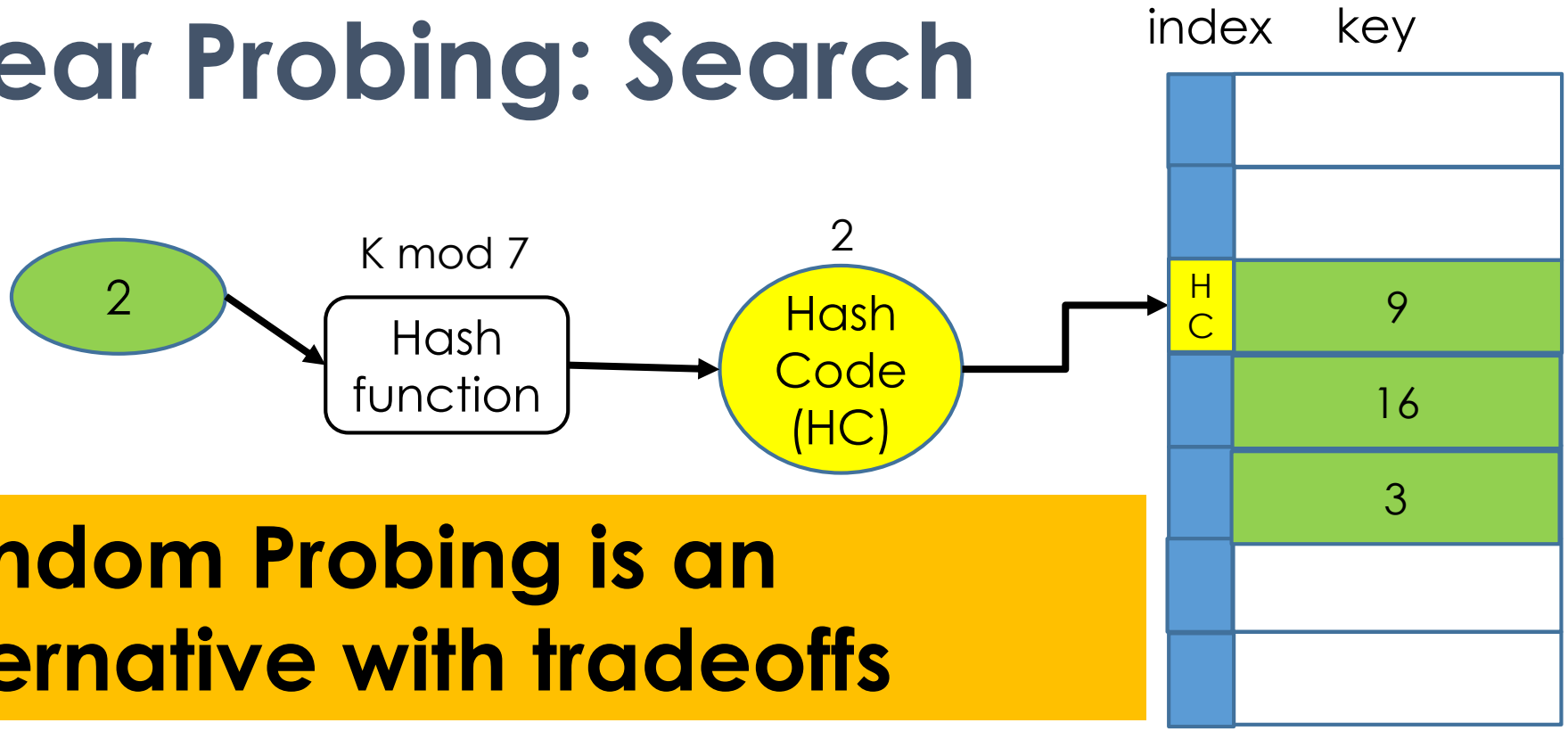


Linear Probing: Search



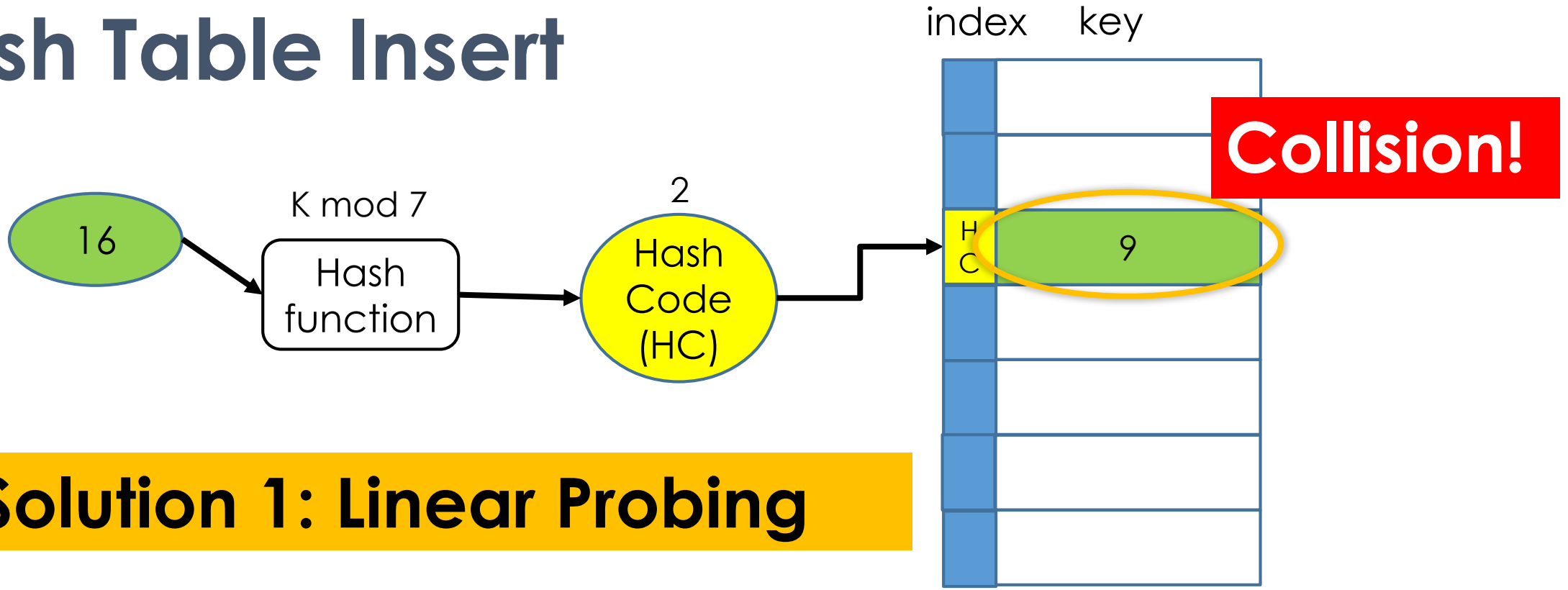
Linear Probing can struggle as the hash table starts getting full

Linear Probing: Search



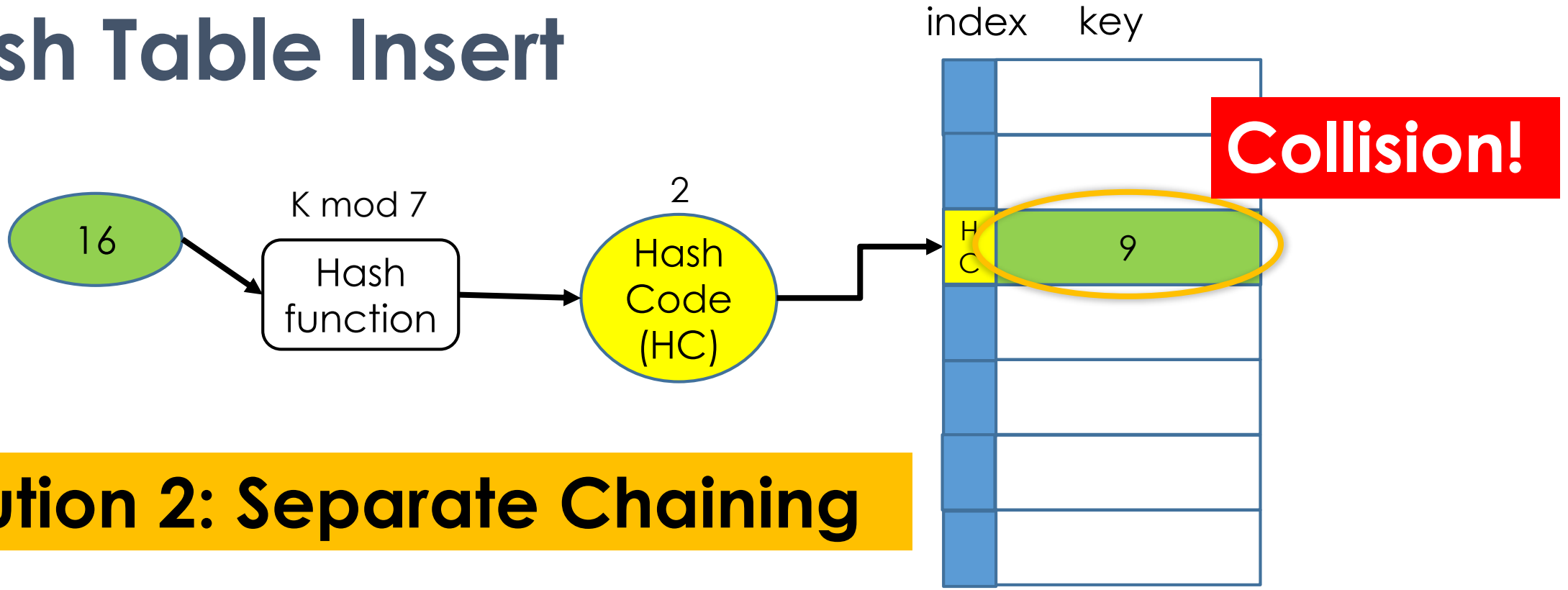
Random Probing is an alternative with tradeoffs

Hash Table Insert



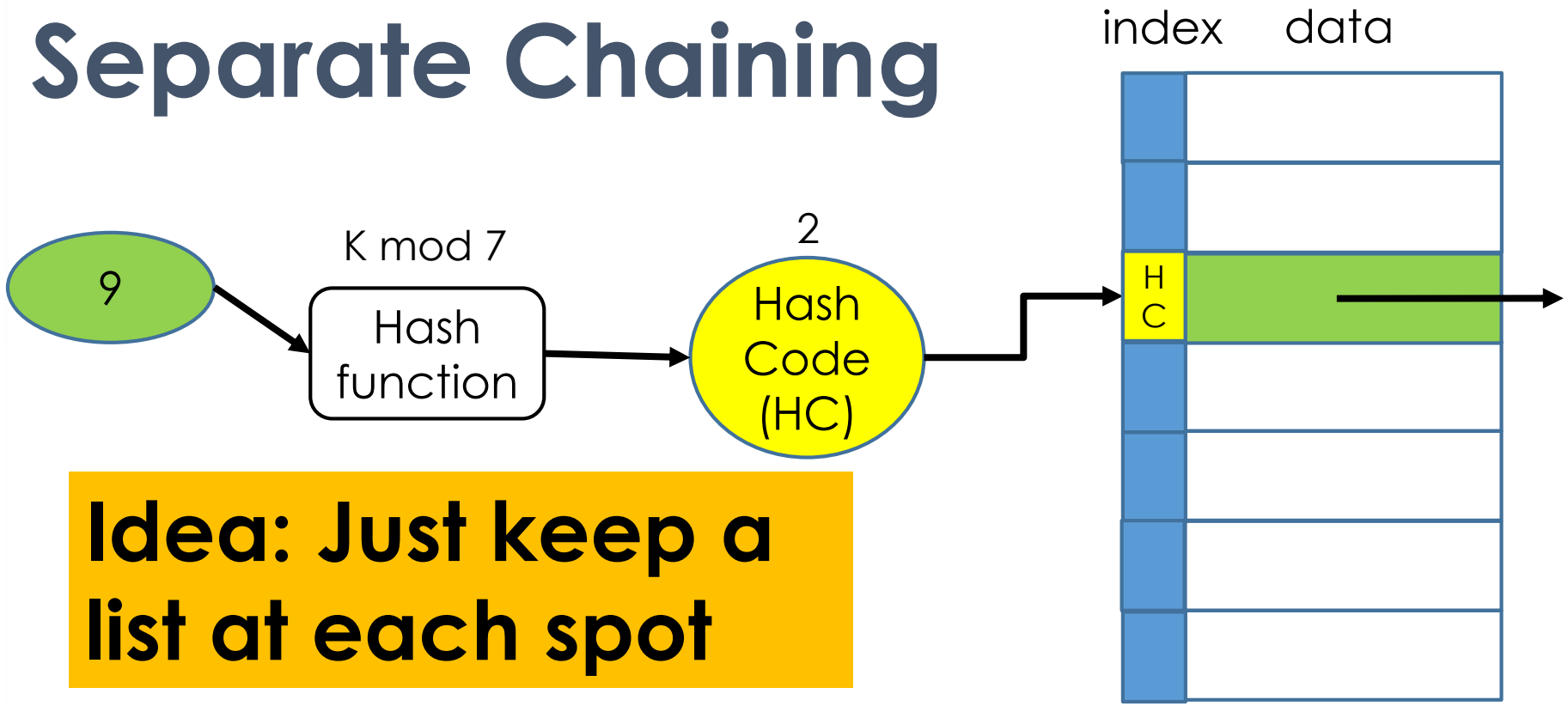
Solution 1: Linear Probing

Hash Table Insert



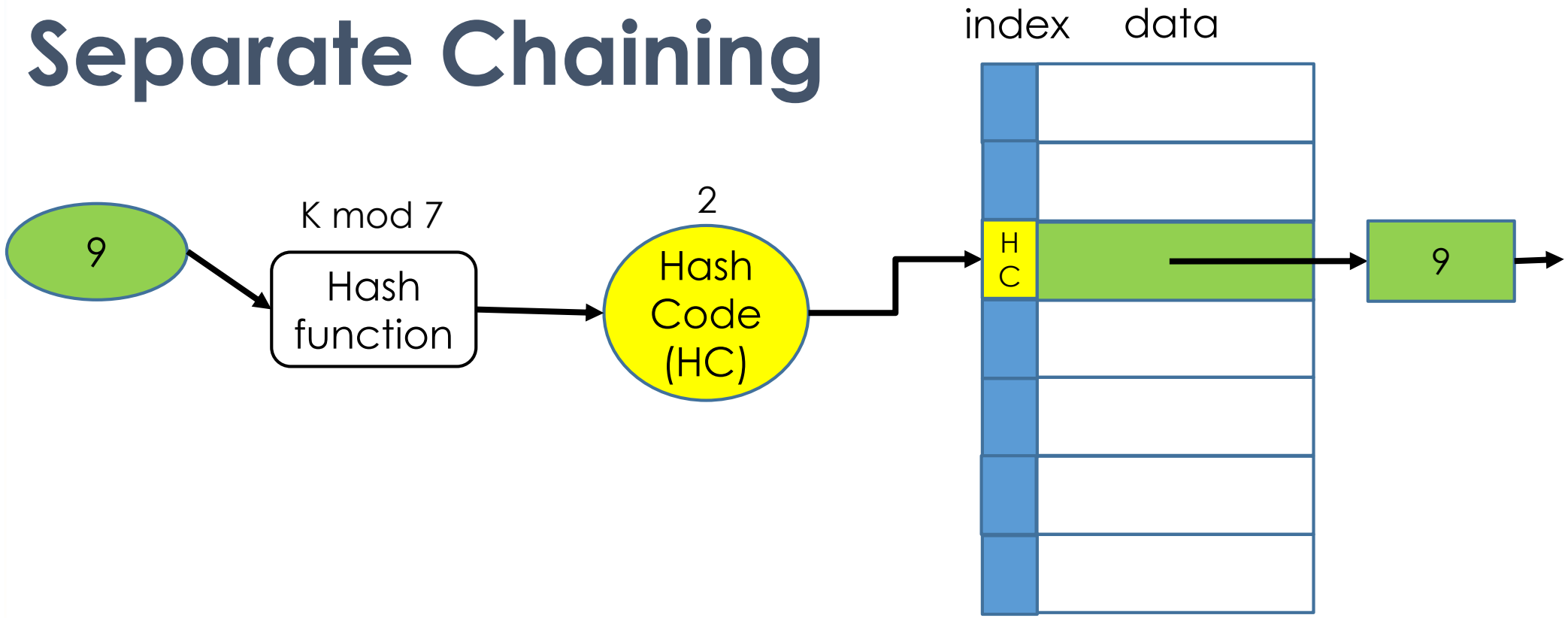
Solution 2: Separate Chaining

Separate Chaining

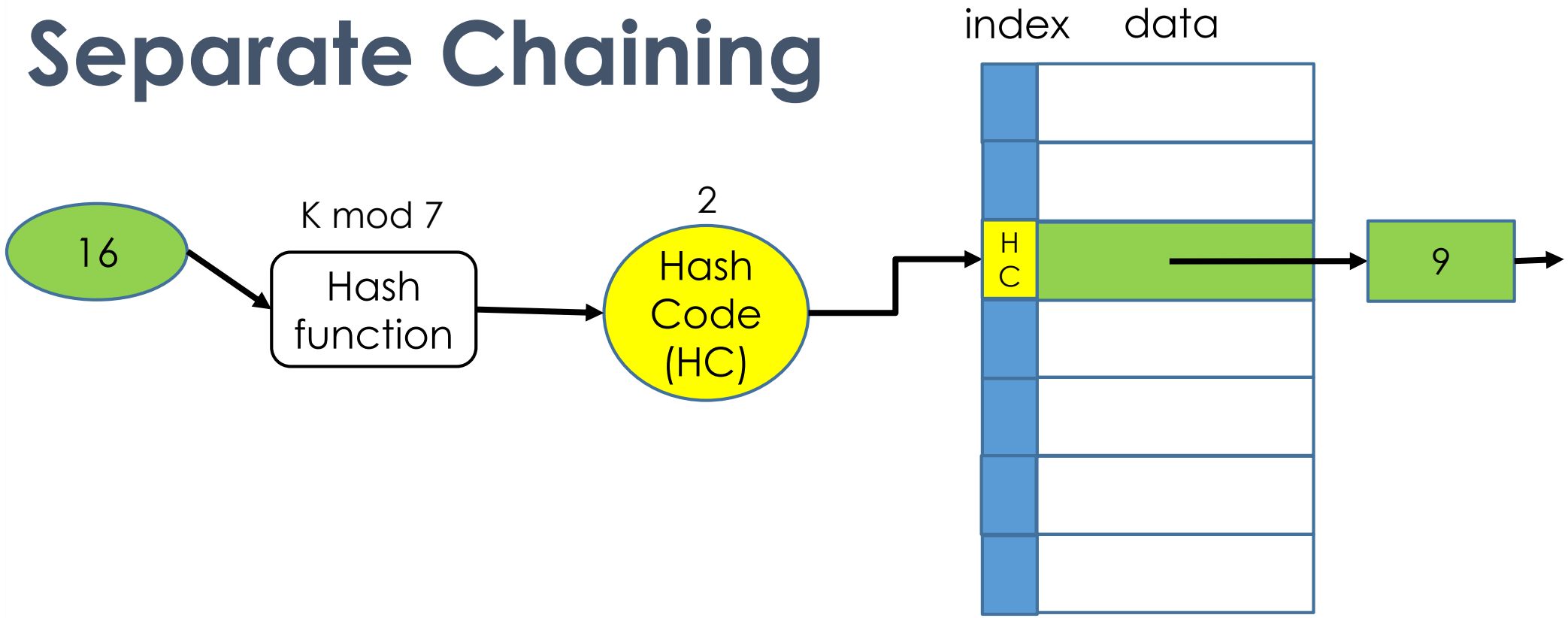


Idea: Just keep a list at each spot

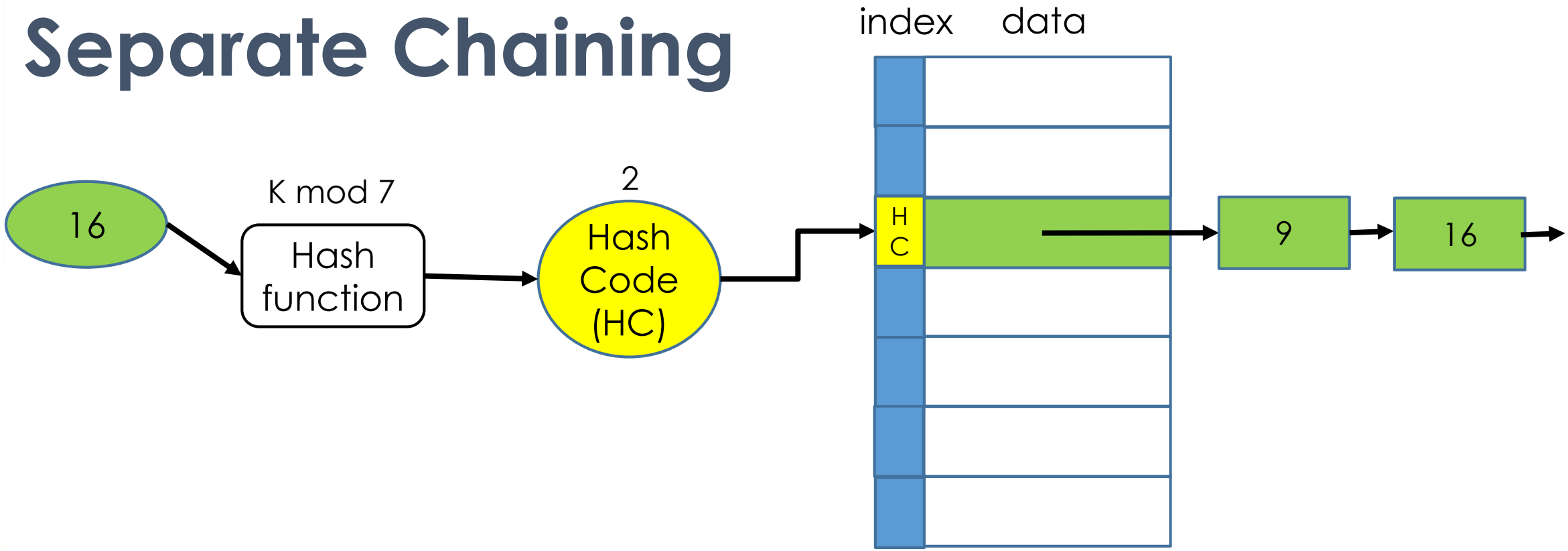
Separate Chaining



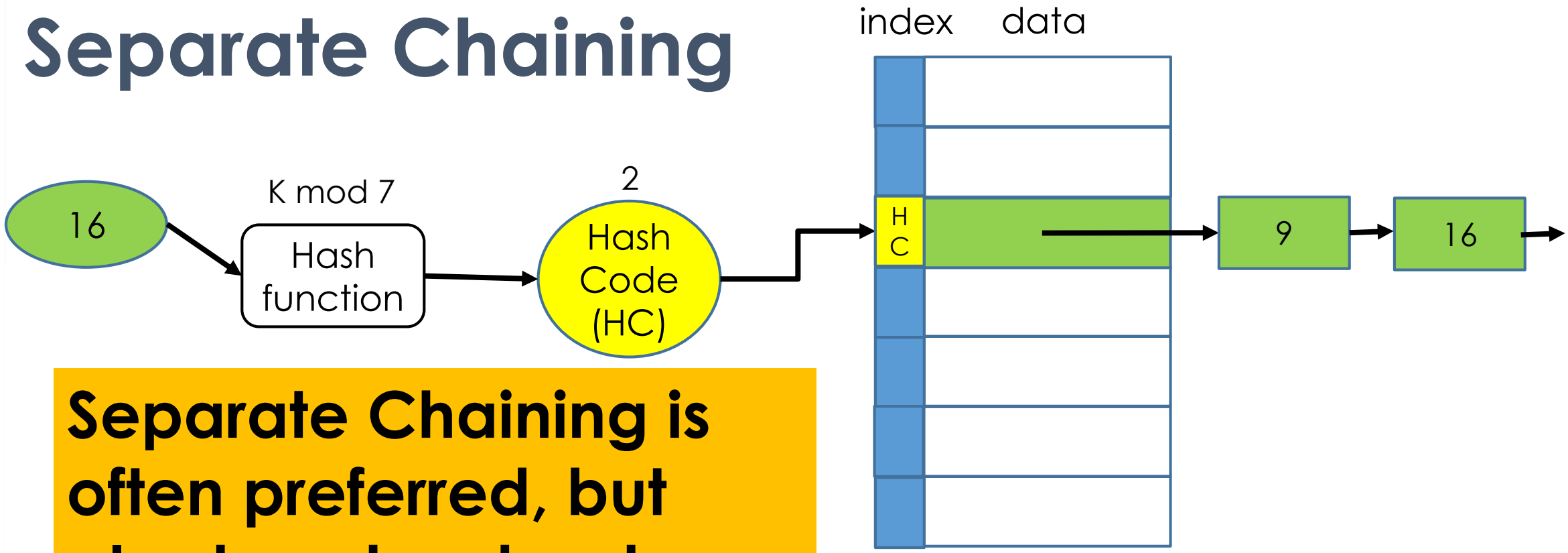
Separate Chaining



Separate Chaining



Separate Chaining



Separate Chaining is often preferred, but also has drawbacks

Challenge 1: Resizing

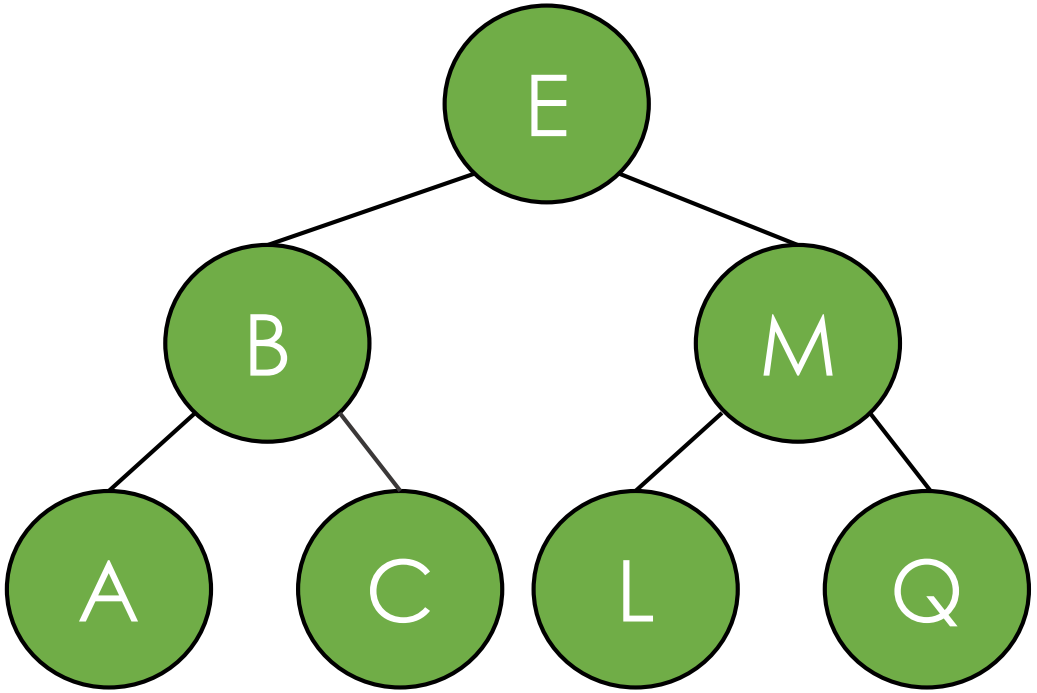
When a hash table gets too full, the best thing to do is resize it.

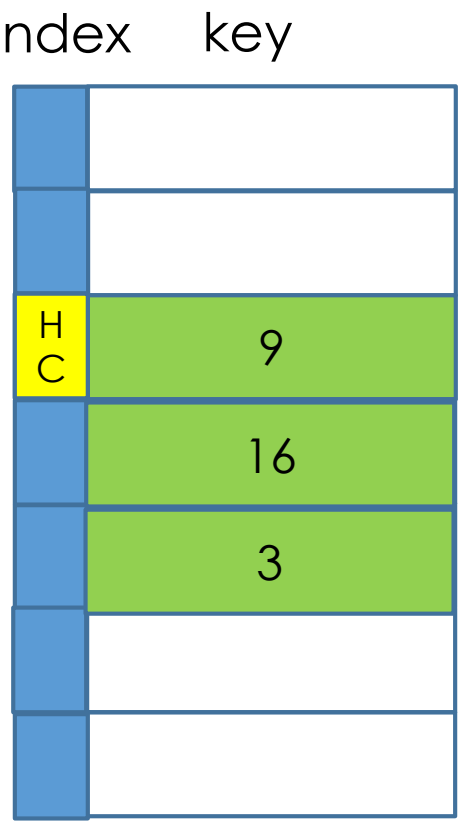
Challenge 1: Resizing

When a hash table gets too full, the best thing to do is resize it.

Requires you create a new table, new hash function, and reinsert everything!

Challenge 2: Ordering data





Hash Table Implications

Average: O(1) lookup, insert, and remove

Resizing costs
No data ordering

http://docs.oracle.com/javase/7/docs/api/java/util/HashSet.html

Summary

- We've looked at solving collisions with:
  - Linear Probing
  - Separate Chaining
- Seen additional Hash Table challenges
- Next, we'll look at using Hash Tables in Java