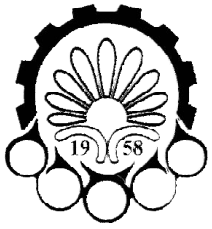


Amirkabir University of Technology
(Tehran Polytechnic)

Department of
Computer Engineering

Data Structure & Algorithms

Hash Tables

Hash Tables - Motivation

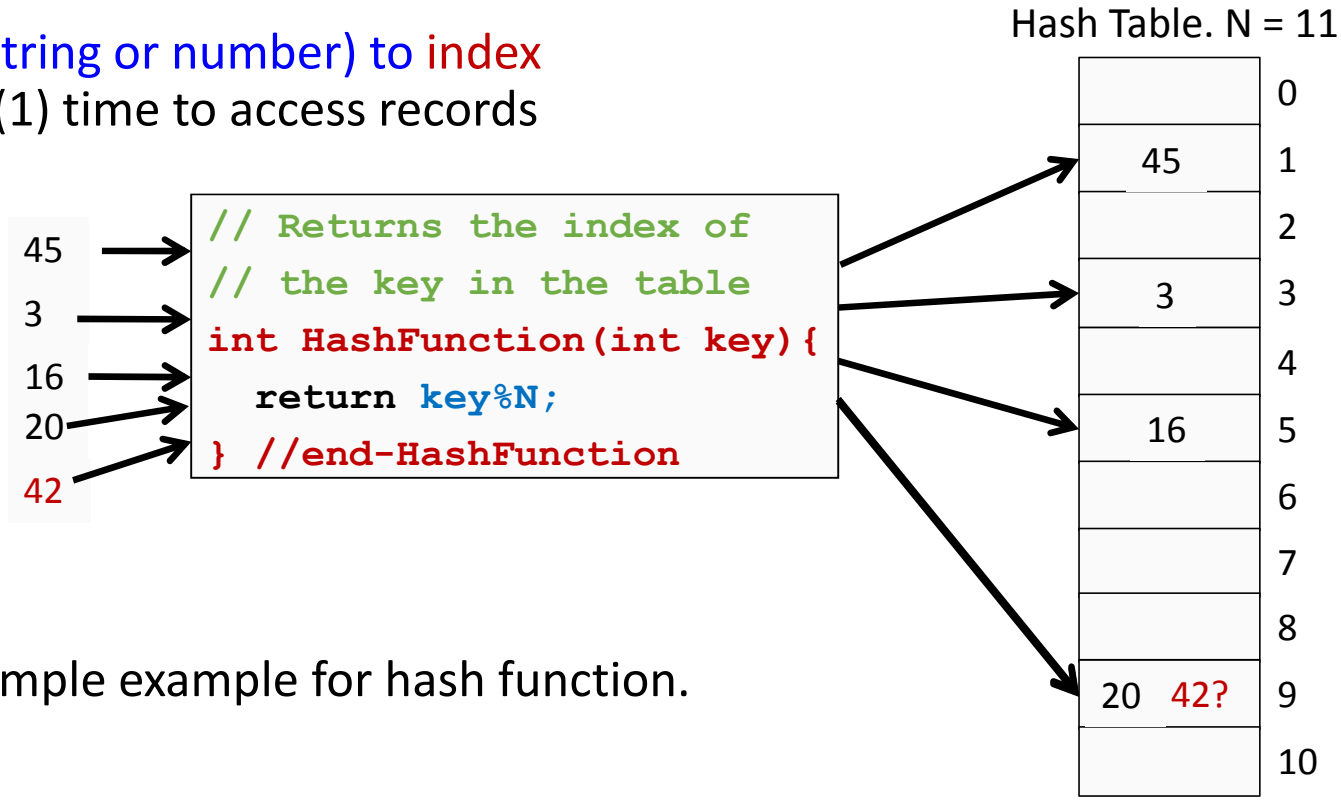
- Consider the problem of storing several (key, value) pairs in a data structure that would support the following operations efficiently
  - Insert(key, value)
  - Delete(key, value)
  - Find(key)

- Data Structures we have looked at so far
  - Arrays – O(1) Insert/Delete, O(N) Search
  - Linked Lists – O(1) Insert/Delete, O(N) Search
  - Search Trees (BST, AVL, Splay) – all ops O(logN)

  Can we make Find/Insert/Delete all O(1)?

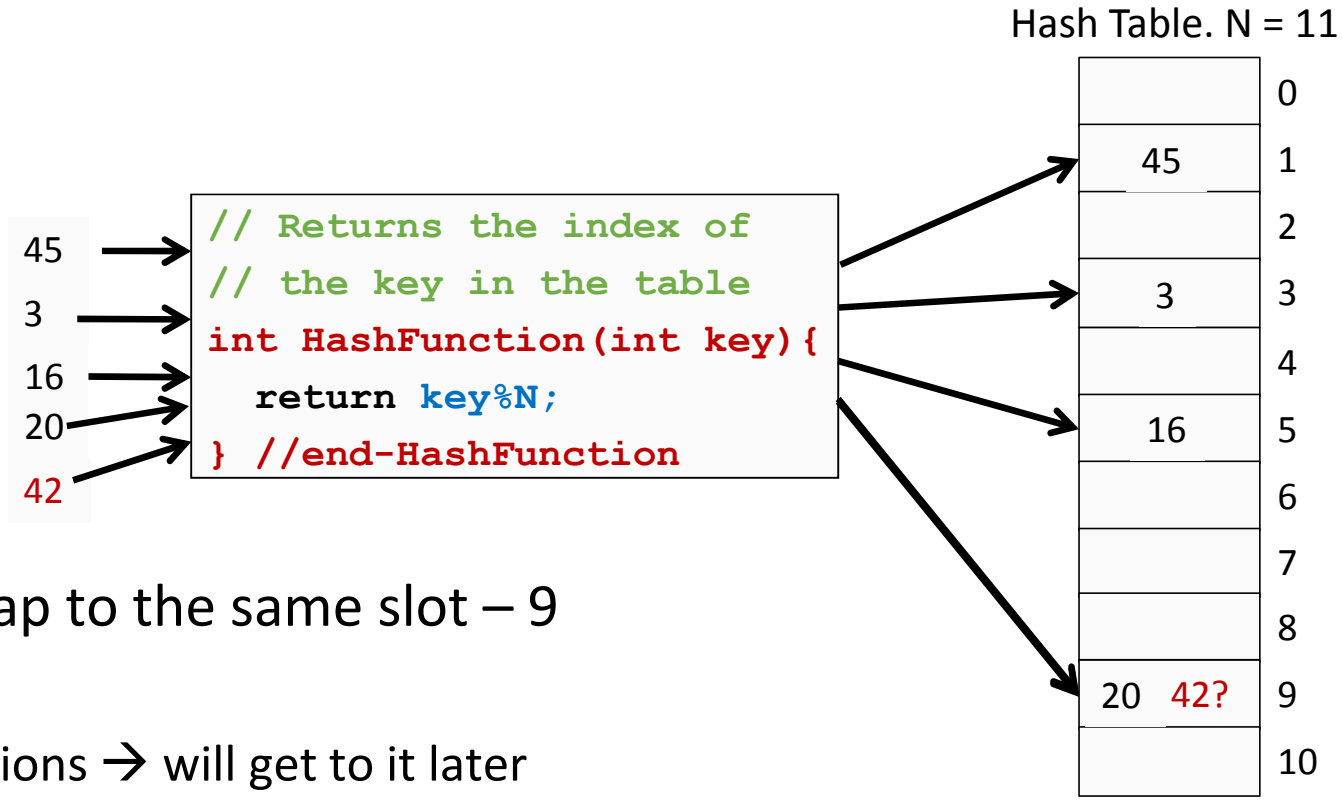
Hash Tables – Main Idea

- Main idea: Use the key (string or number) to index directly into an array – O(1) time to access records



- Attention! This is just a simple example for hash function.

Hash Tables – Main Idea



- Keys 20 & 42 both map to the same slot – 9
  - Called a collision!
  - Need to handle collisions → will get to it later

Hash Functions

- A function that converts a given big phone number to a small practical integer value. The mapped integer value is used as an index in the hash table. In simple terms, a hash function maps a big number or string to a small integer that can be used as the index in the hash table.

Properties of a Good Hash Function

- Should be efficiently computable – O(1) time
- Should hash evenly throughout hash table
- Should utilize all slots in the table
- Should minimize collisions

Hash Table Size

- We need to make sure that Hash Table is big enough for all keys and that it facilitates the Hash Function's job to evenly distribute the keys
  - What if TableSize is 10 and all keys end in 0?
    - All keys would map to the same slot!
  - Need to pick TableSize carefully
    - typically, a prime number is chosen

Collisions and their Resolution

- A collision occurs when two different keys hash to the same value
  - E.g. For *TableSize* = 17, the keys 18 and 35 hash to the same value
  - 18 mod 17 = 1 and 35 mod 17 = 1

- Cannot store both data records in the same slot in array!

- Two different methods for collision resolution:
  - **Separate Chaining:** Use data structure (such as a linked list) to store multiple items that hash to the same slot

  - **Open addressing (or probing):** search for empty slots using a second function and store item in first empty slot that is found

We don't read this methods in details.