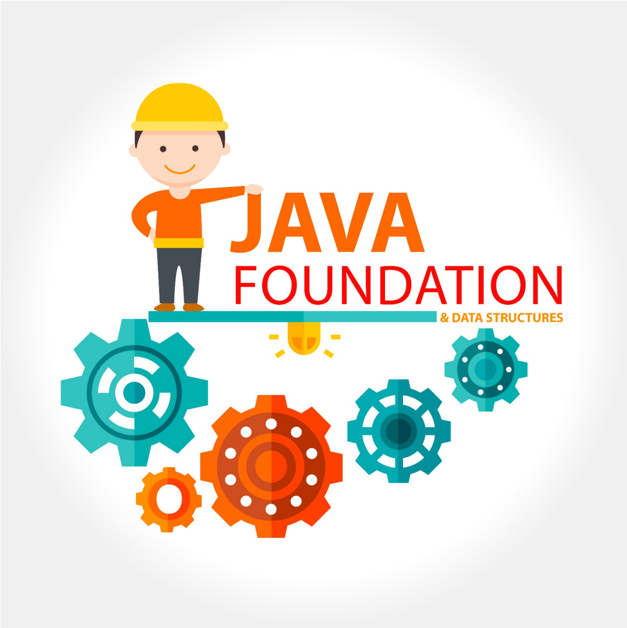
Nucleus

Java Foundation & Data Structures

Lecture 21 : Hashmaps



Friday, 14 July 17

Doubts from Last Class ?

Assignment ?

Maps

Map

```
class Map{
        // accessor methods
        int size();
        boolean isEmpty();
        Object get(Object key);
        // update methods
        void put(Object key, Object value);
        void remove(Object key) ;
        Object[] keys();
        Object[] values();
}
```

Key Functions in map

- Find
- Add
- Remove

Implement a Map using ?

- Unsorted arrays
- Sorted arrays
- Linked List

Any Other Options?

Hashtables

Components of Hash Table

- Bucket Array
- Hash Function –
  - hash code
  - compression function

Overview

- Hash Table Data Structure : Purpose

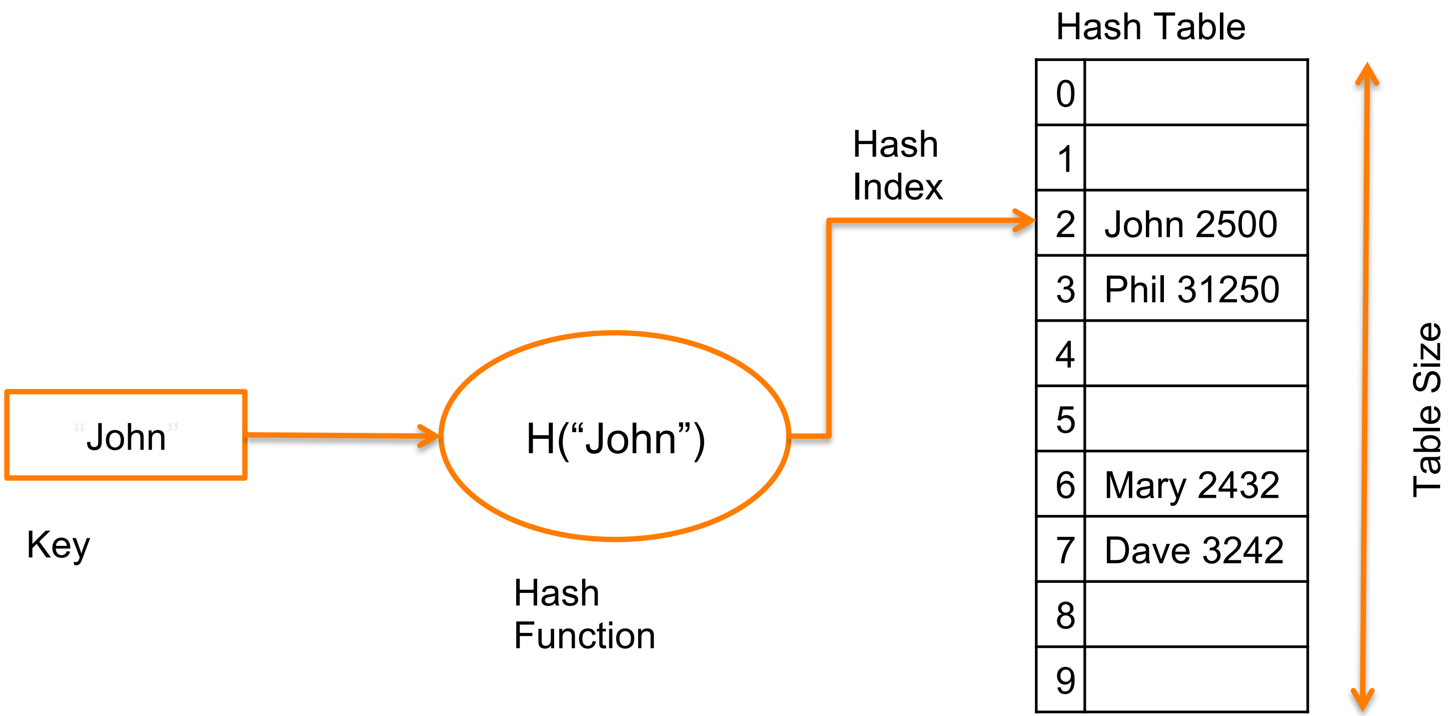
  - To support insertion, deletion and search in average-case constant time

    - Assumption: Order of elements irrelevant
    - data structure *not* useful for if you want to maintain and retrieve some kind of an order of the elements

- Hash function

  - Hash[“string key”] ==> integer value

Key Components



0	
1	
2	John 2500
3	Phil 31250
4	
5	
6	Mary 2432
7	Dave 3242
8	
9	

Hash Table Operations

- Insert – T[h(key)] = value;
- Delete – T[h(key)] = null;
- Search – return T[h(key)];

What happens if h(“john”) == h(“joe”)

Collision!

Factors!

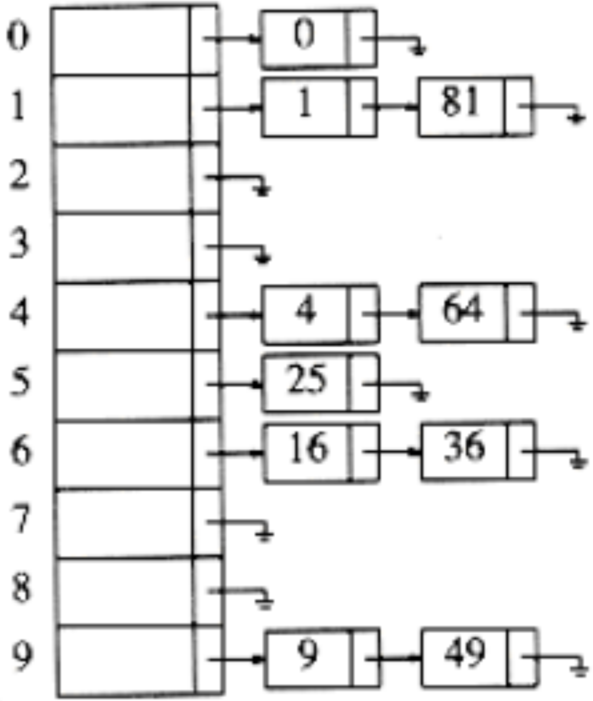
- Hash Function
- Table Size – usually fixed at the beginning
- Collision handling Scheme

How to handle Collisions?

- Open Hashing – Separate Chaining
- Closed Hashing – Open Addressing
    - Linear Probing
    - Quadratic Probing
- Double Hashing

Separate Chaining

- Implemented using Linked Lists.

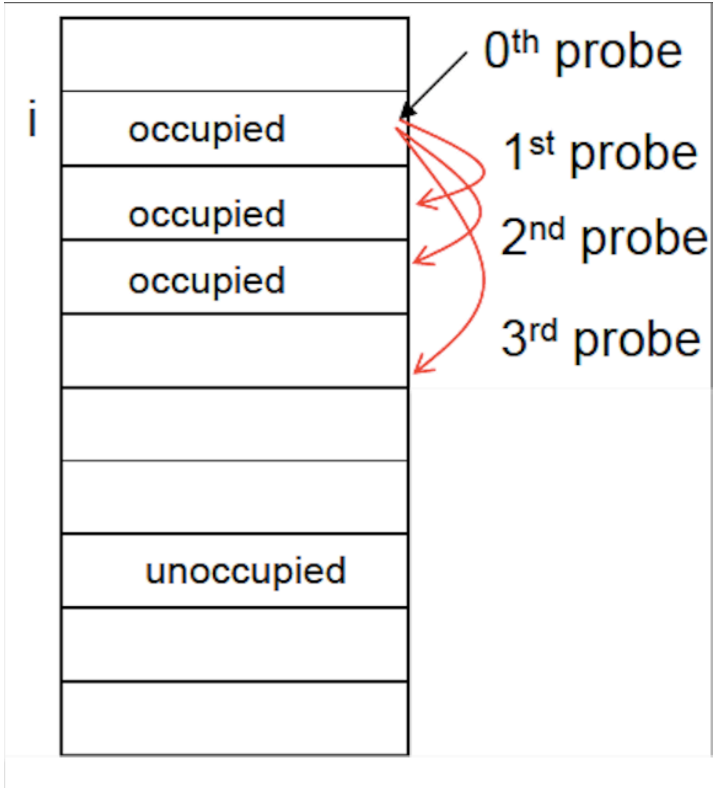


Running time for separate chaining?

- Find
- Add
- Remove

Linear Probing

f(i) = is a linear function of I
For e.g., f(i) = i



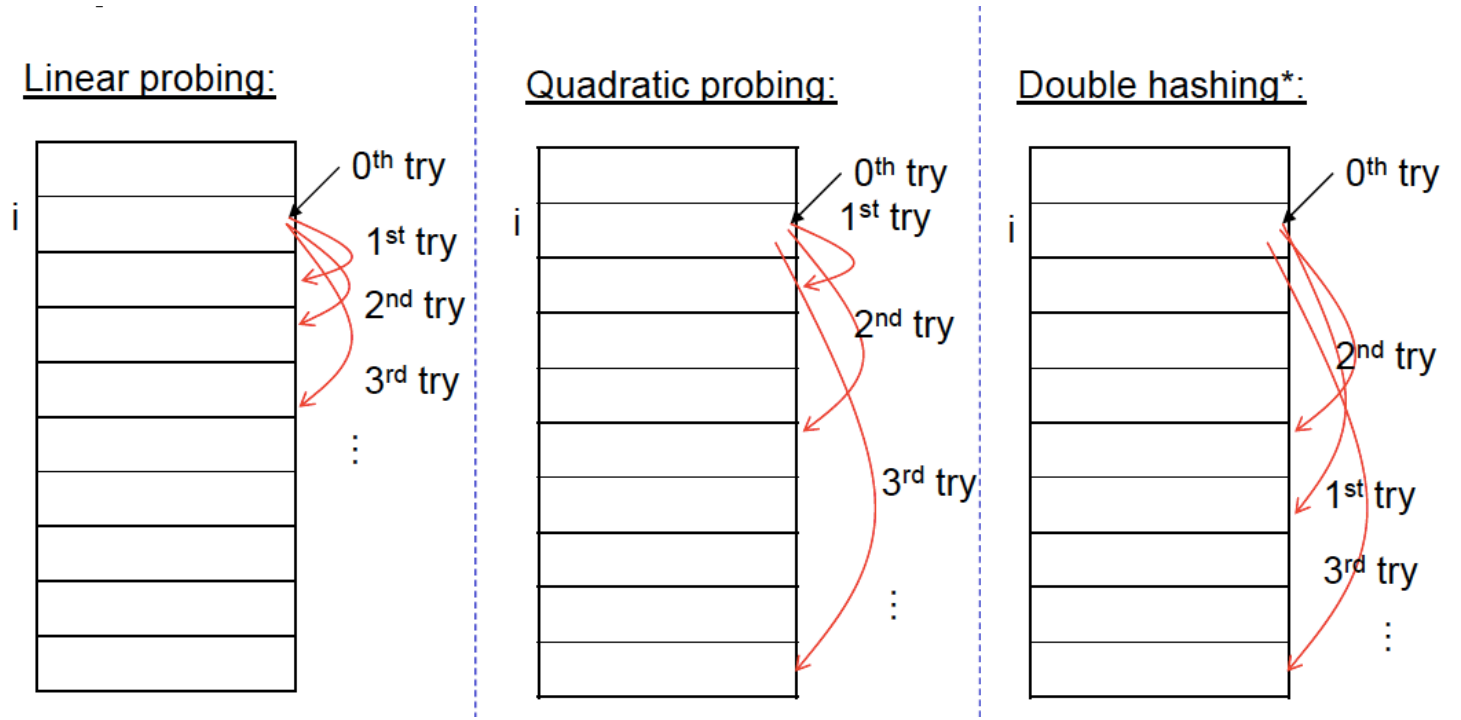
Quadratic Probing

- Avoids primary clustering
- f(i) is quadratic in i : f(i)= i^2

Double Hashing

- Use a second hash function for all tries of i other than 0: f(i) = i * h2(x)

Probing Techniques



Load factor and rehashing

Lets see some examples !

Lets discuss few problems

- Find intersection of two sorted arrays. What about unsorted arrays?

Your Turn

- Remove Duplicates from an array
- Find pairs of elements in an array which sum to zero
- Union and Intersection of two Linked Lists

Your turn

- Print a Binary Tree in Vertical Order

```
       1
     /     \
    2       3
  / \      / \
4    5   6    7
       \   \
        8    9
```

The output of print this tree vertically will be -

```
4
2
1 5 6
3 8 9
7
```

Lets see implementation!

CODING NINJAS

Thank you

Nidhi Agarwal
nidhi@codingninjas.in