

Dictionary

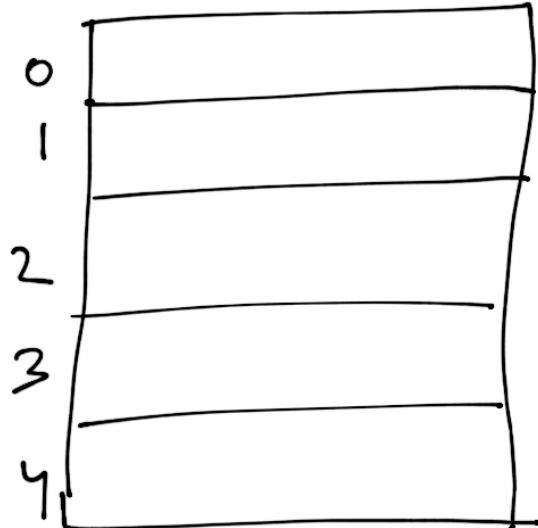
- ADT
- Maintain a set of items, each with a key
 - insert(item)
 - delete(item)
 - search(key)

Motivation

- database
- password/login
- compilers & interpreters
- network router
- String commonalities

Direct Access Table

→ Array where index can work as key



Badness

- ① Keys may not be non-negative integers
- ② gigantic memory hog
prehash : Keys to non-negative integers

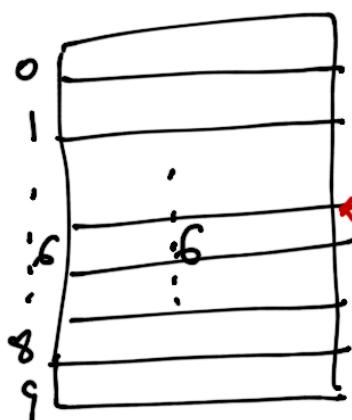
Hashing

→ Searching Technique
→ $O(1)$

$$\text{Key} = 6, 26$$

$$h(k_i) = k_i \% m$$

$$\text{Size of Hash table (m)} = 10 \quad h(6) = 6 \% 10 \\ = 6$$



Collision

- ① Open Hashing (closed Addressing)
 - ↳ Chaining
- ② Closed Hashing (Open Addressing)
 - Linear Probing
 - Quadratic Probing
 - Double Hashing

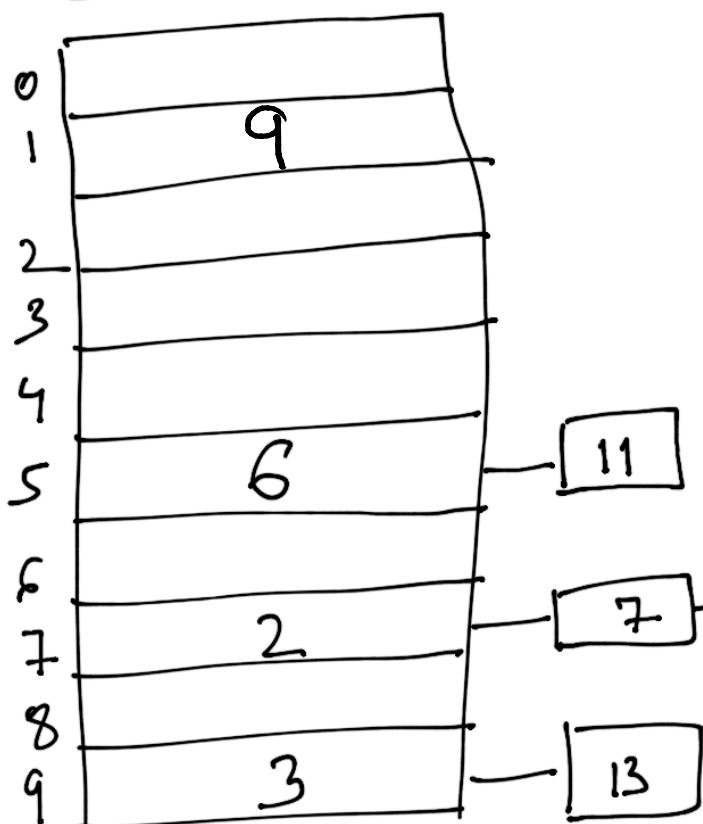
Chaining

$$K = 3, 2, 9, 6, 11, 13, 7, 12$$

$$h(K) = 2K + 3, \boxed{m = 10}$$

Use Division method $K \% m$

Closed Addressing to store the value



Key	Location	$(K \% m)$
3	$(2 \times 3 + 3) \% 10 = 9$	9
2	$(2 \times 2 + 3) \% 10 = 7$	7
9	$(2 \times 9 + 3) \% 10 = 1$	1
6	$(2 \times 6 + 3) \% 10 = 5$	5
11	$(2 \times 11 + 3) \% 10 = 5$	5
12	$(2 \times 13 + 3) \% 10 = 9$	9
7	$(2 \times 7 + 3) \% 10 = 2$	2
12	$(2 \times 12 + 3) \% 10 = 7$	7

Linear Probing

$$h(k) = 2k+3, k=3, 2, 9, 6, 11, 13, 7, 12$$

$m=10$

Use Division method &

Open Addressing to store the value

* Insert k_i at first free location from $\frac{(h(i))}{\%} \cdot m$
 where $i = 0, 1, \dots, 9$

0	13
1	9
2	12
3	
4	
5	6
6	11
7	2
8	7
9	3

Key	Location	Probes
3	$(2 \times 3 + 3) \% \cdot 10 = 9$	1
2	$(2 \times 2 + 3) \% \cdot 10 = 7$	1
9	$(2 \times 9 + 3) \% \cdot 10 = 1$	1
6	$(2 \times 6 + 3) \% \cdot 10 = 5$	1
11	$(2 \times 11 + 3) \% \cdot 10 = 5$	2
13	$(2 \times 13 + 3) \% \cdot 10 = 9$	2
7	$(2 \times 7 + 3) \% \cdot 10 = 7$	2
12	$(2 \times 12 + 3) \% \cdot 10 = 7$	6

Order of Elements

13, 9, 12, -, -, 6, 11, 2,
 7, 3

Quadratic Probing

Insert K_i at first free location from $(u+i^2) \% m$ where $i=0, 1, \dots, m-1$

$$K = 3, 2, 9, 6, 11, 13, 7, 12 \quad m = 10, h(k) = 2k+3$$

Use Division method & Quadratic Probing to share these values

0	13
1	9
2	
3	12
4	
5	6
6	11
7	2
8	7
9	3

for 11
 $u=5$
 $(5+0) \% 10 \rightarrow 5$
 $= 5$
 $(5+1) \% 10 \rightarrow 6$
 $= 6$
for 13
 $u=9$
 $(9+0) \% 10 = 9$
 $(9+1) \% 10 = 0$

Key	Location(u)	Probe
3	$(2 \times 3 + 3) \% 10 = 9$	1
2	$(2 \times 2 + 3) \% 10 = 7$	1
9	$(2 \times 9 + 3) \% 10 = 1$	1
6	$(2 \times 6 + 3) \% 10 = 5$	1
11	$(2 \times 11 + 3) \% 10 = 5$	2
13	$(2 \times 13 + 3) \% 10 = 9$	2
7	$(2 \times 7 + 3) \% 10 = 7$	2
12	$(2 \times 12 + 3) \% 10 = 7$	5

Order: 13, 9, -, 12, -, 6, 11, 2, 7, 3

Double Hashing

$$K = 3, 2, 9, 6, 11, 13, 7, 12$$

$$h_1(K) = 2K + 3, m = 10$$

Use division method & double hashing to insert these elements

$$h_2(K) = 3K + 1$$

Insert k_i at first free location from
 $(u+v*i) \% m, i = 0 \text{ to } m-1, v = h_2(k) \% m$

0	
1	9
2	
3	11
4	12
5	6
6	
7	2
8	
9	3

Key	Location(u)	v	Probe
3	$(2 \times 3 + 3) \% 10 = 9$	-	1
2	$(2 \times 2 + 3) \% 10 = 7$	-	1
9	$(2 \times 9 + 3) \% 10 = 1$	-	1
6	$(2 \times 6 + 3) \% 10 = 5$	-	1
11	$(2 \times 11 + 3) \% 10 = 5$	4	3
13	$(2 \times 3 + 3) \% 10 = 9$	x	x
7	$(2 \times 7 + 3) \% 10 = 7$	2	x
12	$(2 \times 12 + 3) \% 10 = 7$	7	2

for 11

$u = 5$

$v = (3k+1) \% m$

$= (3 \times 11 + 1) \% 10$

$= 34 \% 10$

$= 4$

$(u+v*i) \% m$

$= (5 + 4 * 0) \% 10$

$= 5 \% 10 = 5$

$(5 + 4 * 1) \% 10$

$= 9 \% 10 = 1$

$(5 + 4 * 2) \% 10$

$= 13 \% 10 = 3$

Good Hash functions

1. Easy to compute $H(n) O(1)$
2. Even distribution

$$H(n) = n \bmod 10$$

Keys : 10, 20, 100, 200, 1000

$$H(n) = n \bmod 10$$

This resulting into more collisions.

$$H'(n) = n \bmod 7$$

3. Maintain loadfactor less

λ = Load factor

$$\lambda = \frac{N}{T} \leftarrow \begin{array}{l} \text{No. of Elements} \\ \text{in the hashtable} \end{array}$$

$$T \leftarrow \text{Table size}$$

0	10, 20, 100, 200, 1000
1	
2	100
3	10
4	200
5	
6	20
7	

→ Low ' λ ' factor means less collisions

Eg: given a hash table with 25 slots that stores 2000 elements,

$$\text{Load factor}(\lambda) = \frac{2000}{25} = 80$$