HASHING

Hashing is a technique used for performing insertion, deletion and find operation in constant average time of **O(1)**

Hashing is the process of transforming any given key or a string of characters into another value.

For Example,

Using Unsorted sequential array, Insertion=O(1), deletion=O(n) & find=O(n)

Using Sorted sequential array, Insertion =O(n), deletion=O(n) & find=O(log n)

Searching	Best case	Average case	Worst case
Linear Search	O(1)	O(n)	O(n)
Binary Search	O(1)	O(log n)	O(log n)
Hashing	O(1)	O(1)	O(n)

if we do not know the position of element, then linear search need o(n) complexity and Binary search need O(log n) complexity.

if we know the position of element in the array, then we could quickly access it.

HASH TABLE

- A hash table is data structure for storing a set of items, so that we can quickly perform searching, insertions and deletions.
- A hash table is normally created with a certain number of buckets or storing locations.
- A hash table data structure is a fixed size array containing keys.

- Data element or elements in a hash tables is identified by a key or keys.

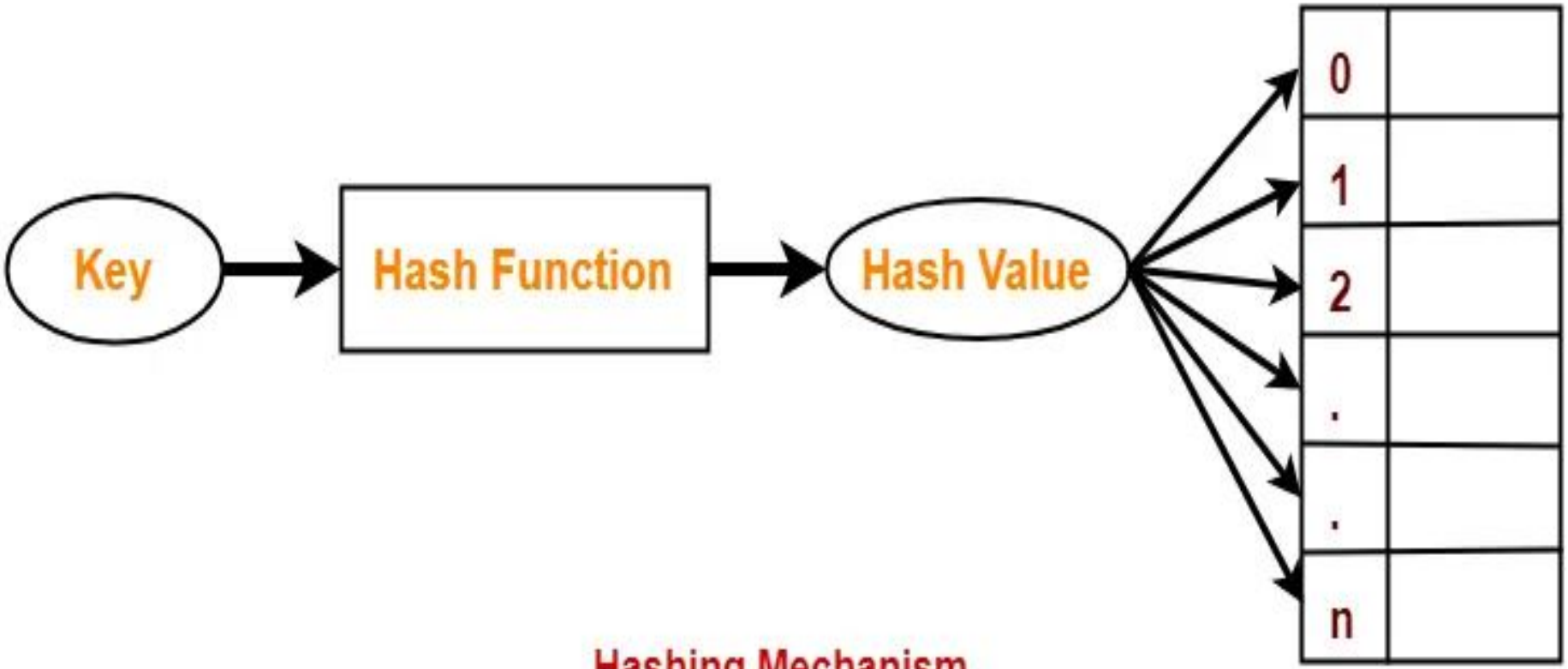
- Each key is mapped into some number in the range 0 to m-1, where m is the size of the hash table.

HASH FUNCTION

hash function is a function h(k), which transform a key ‘K’ into an address in a hash table.

Keys

A record may consist of several fields of data, one of which is the key.



Hashing Mechanism

- In order to insert a new record, the **key** is **converted to** an array **index**.
- The index is called the **hash value** of the key.
- Commonly used method to find hash value is (key) mod (size of hash table)

= 506643502 mod 500 = 2

store at index 2

The following are the important Hash functions

(1) The Division Method

is defined as

$$h(k) = k \bmod M$$

where K is the integer key & M is the size of Hash table.

example:

Hash table using Mod M=9

Keys = 5, 17, 37, 20, 42, 3

0	1	2	3	4	5	6	7	8
	37	20	3		5	42		17

Example 2

$M = 10$

What is the hash key for 'Cat'

∴ Cat = 131130 when Converted to ASCII

∴ $X = 131130$

∴ $H(x) = x \bmod M$

$H(131130) = 131130 \bmod 10 = 0$

(2) The folding Method

Partition the key into several parts and Combine arithmetically, (addition (or) subtraction) then truncate. The other hashing functions are used on that.

example

Key = 12345678 $M = 10$

∴ a = 123 b = 456, c = 789

e $H(12345678) = (123+456+789) \bmod 10$

$= 1368 \bmod 10$

$= 8$

e 12345678 is inserted at address 8

(3) Mid Square Method

In this Method a key is multiplied by itself and the address is obtained by selecting an appropriate number of bits or digits from the square. number of bits depend on the size of table.

ex: key = 123456

square = 15241383936

Digit Extraction Method

digits are extracted from the Original key and used as the address

ex: Picking the first, third, & fourth digits from ~~right~~ left.

379452 ⟶ 394

121267 ⟶ 112

378845 ⟶ 388

160252 ⟶ 102

045188 ⟶ 051

Multiplication Method

$$H(x) = (a \times b * c * d * \cdots) \bmod n$$

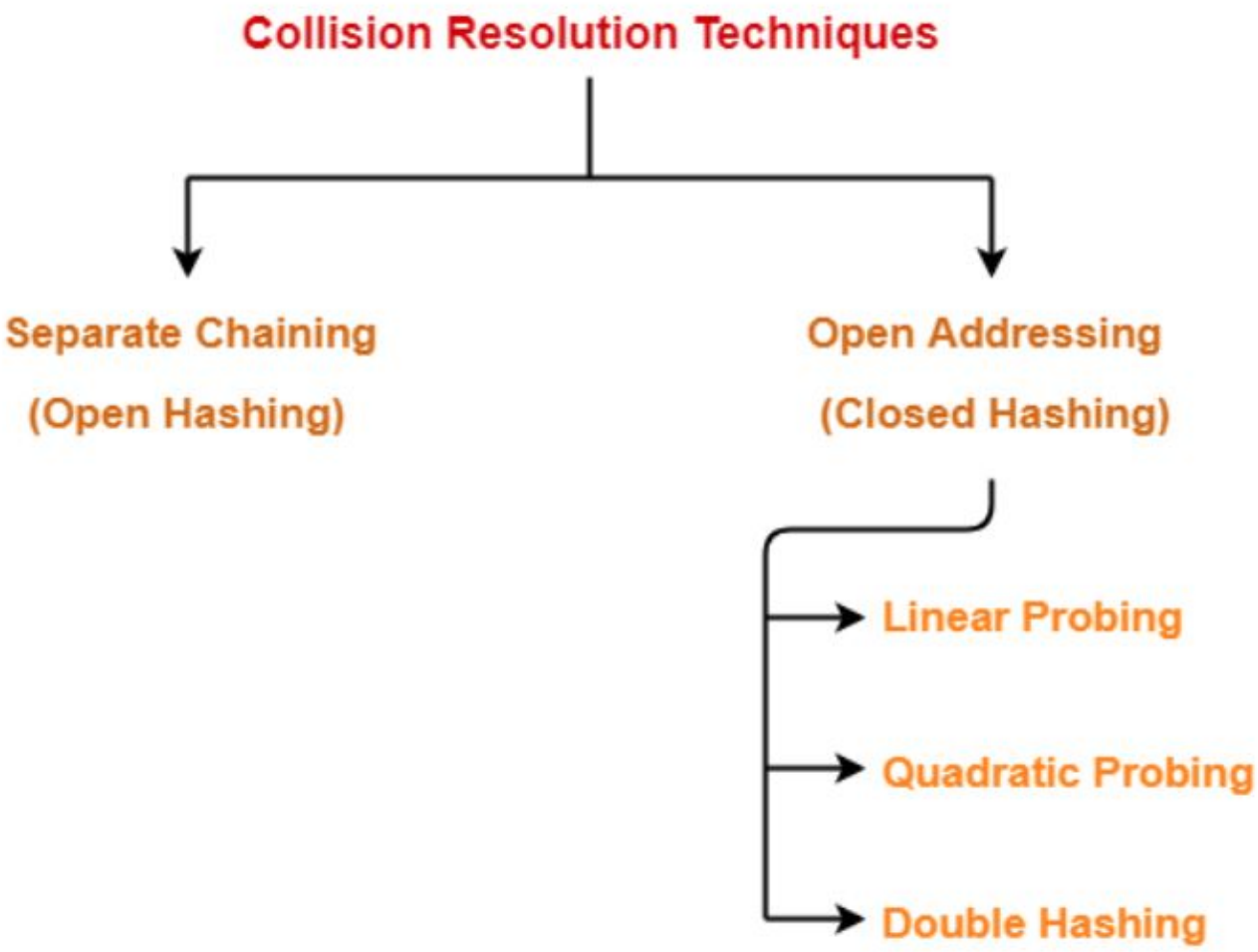
where a, b, c ... are individual digits of the key.

Collision in Hashing-

When the hash value of a key maps to an already occupied bucket of the hash table,
it is called as a **Collision**.

Collision Resolution Techniques-

Collision Resolution Techniques are the techniques used for resolving or handling the collision.


Collision Resolution Techniques
Separate Chaining
(Open Hashing)
Open Addressing
(Closed Hashing)
Linear Probing
Quadratic Probing
Double Hashing

In open hashing, collisions are solved by storing the values in outside of the table .

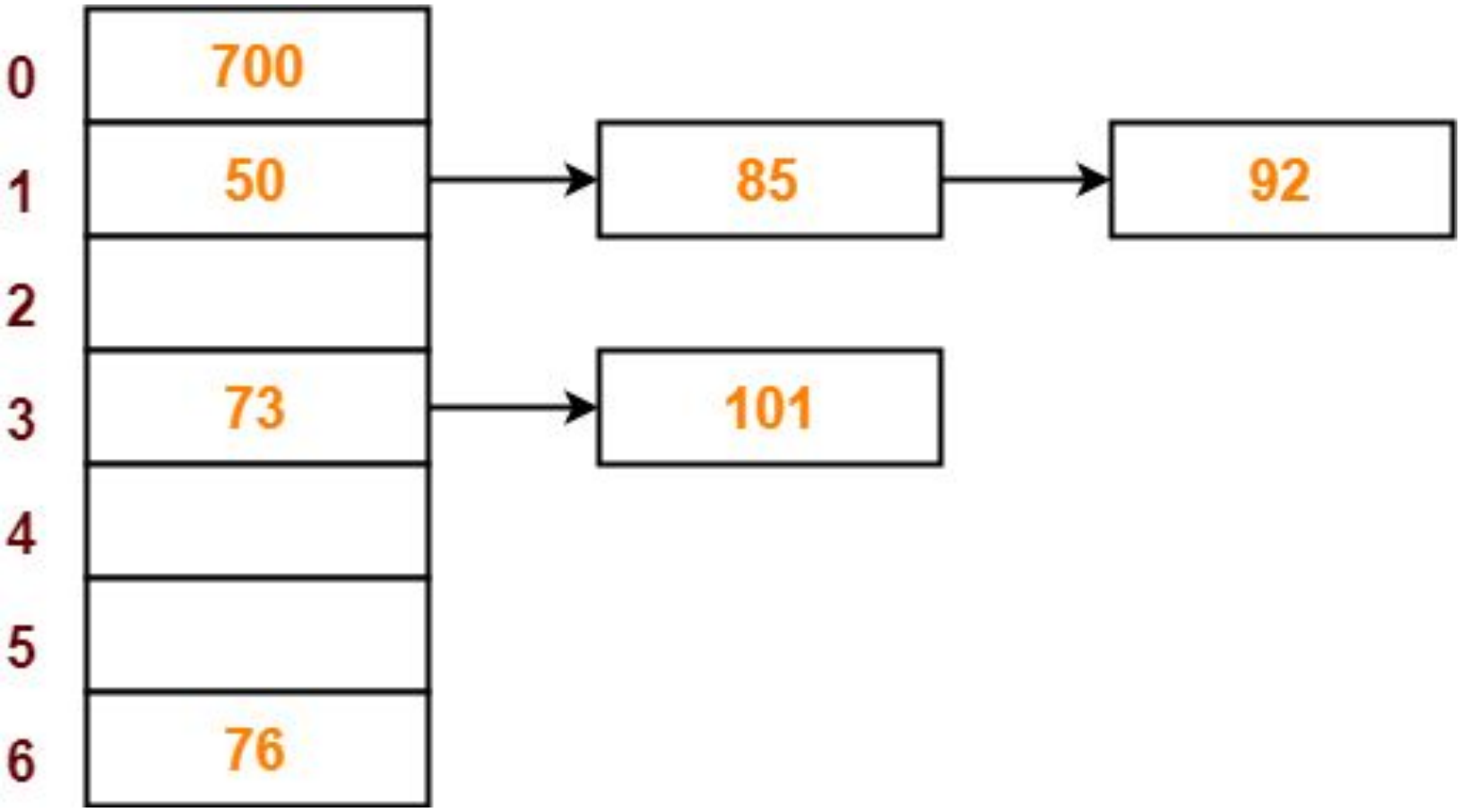
In closed hashing, collisions are solved by storing the values at another slot in the table

SEPARATE CHAINING-

Example 1- Using the hash function ‘key mod 7’,

insert the following sequence of keys in the hash table-
50, 700, 76, 85, 92, 73 and 101

- Draw an empty hash table.
- For the given hash function, the possible range of hash values is [0, 6].
- So, draw an empty hash table consisting of 7 buckets as-


0
700
1
50
85
92
2
3
73
101
4
5
6
76

Example 2:

Array A=3, 2, 9, 6, 11, 13, 7, 12 h(k) = 2k+3 M=10

Use h(k), division method, open addressing to store these values

Key	Location U =(2k+3)mod 10
3	(2x3+3) mod 10 =9
2	7
9	1
6	5
11	**5**
13	**9**
7	**7**
12	**7**

1. Linear Probing-

In linear probing,

- When collision occurs, we linearly probe for the next bucket.
- We keep probing until an empty bucket is found.

Example 2:

Array A=3, 2, 9, 6, 11, 13, 7, 12 h(k) = 2k+3 M=10

Use h(k), division method, open addressing to store these values

Key	Location U =(2k+3)mod 10	Probes
3	(2x3+3) mod 10 =9	1
2	7	1
9	1	1
6	5	1
11	**5**	**2**
13	**9**	**2**
7	**7**	**2**
12	**7**	**6**

Insert Ki at first free location from (U+i)%m, where i=0 to m-1

2. Quadratic Probing-

In quadratic probing,

- When collision occurs, we probe for i^2'th bucket in i^{th} iteration.
- We keep probing until an empty bucket is found.

Example 2:

Array A=3, 2, 9, 6, 11, 13, 7, 12 h(k) = 2k+3 M=10

Use h(k), division method, Quadratic closed addressing to store these values

Key	**Location U =(2k+3)mod 10**	**Probes**
3	(2x3+3) mod 10 =9	1
2	7	1
9	1	1
6	5	1
11	**5**	**2**
13	**9**	**2**
7	**7**	**2**
12	**7**	**5**

Insert Ki at first free location from $(U+i^2)\%m$, where i=0 to m-1

3. Double Hashing

Example 2: Array A=3, 2, 9, 6, 11, 13, 7, 12 h1(k) = 2k+3,h2(k)=3k+1 V= h2(k) % M M=10

Use h(k), division method, Quadratic open addressing to store these values

Key	Location U =(2k+3)mod 10	V	Probes
3	(2x3+3) mod 10 =9	-	1
2	7	-	1
9	1	-	1
6	5	-	1
11	**5**	4	**3**
13	**9**	0	**x**
7	**7**	2	**x**
12	**7**	7	**2**

Insert Ki at first free location from (U+V x i)%m, where i=0 to m-1