Introduction

- Runtime of linear search and linked list O(n)
- Runtime of binary search and binary tree is O(log(n)) which is faster than linear search.
- Can we perform the searching process in O(1). Yes, its possible by hashing.

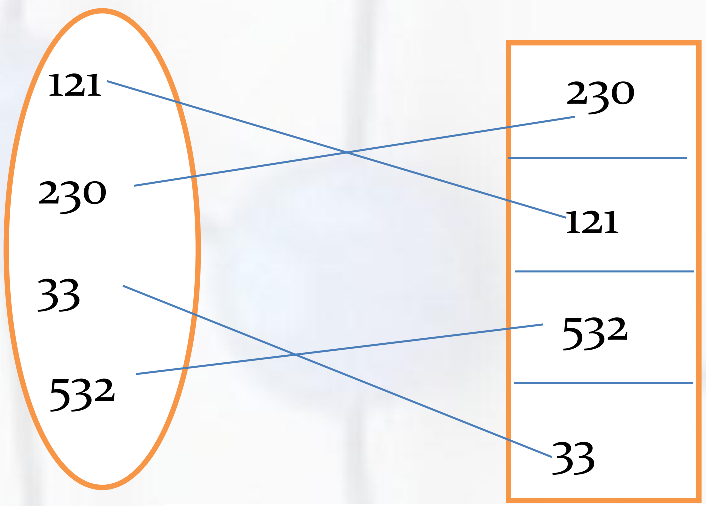
$$O(1) < O(\log(n)) < O(n)$$

Hashing

DEFINITION:

- The processing of mapping keys to appropriate location (indices) in a hash table is called **Hashing.**
- **Hash function** is a mathematical function used in hashing concept which gives the appropriate value of the index of hash table in which a key should be stored.

Example:



Key Space

Hash Table

Hash Function

$h(x) = x \bmod 10$

Hashing

- Hash table should consist of 2 types of values:
  Key Value and **Sentivel Value** (For identification of available space)
- Initially hash table is filled with -1 not with garbage value.

Example:

List of elements : 34, 56, 79, 93, 10, 42, 81, 25

10	81	42	93	34	25	56	-1	-1	79
0	1	2	3	4	5	6	7	8	9

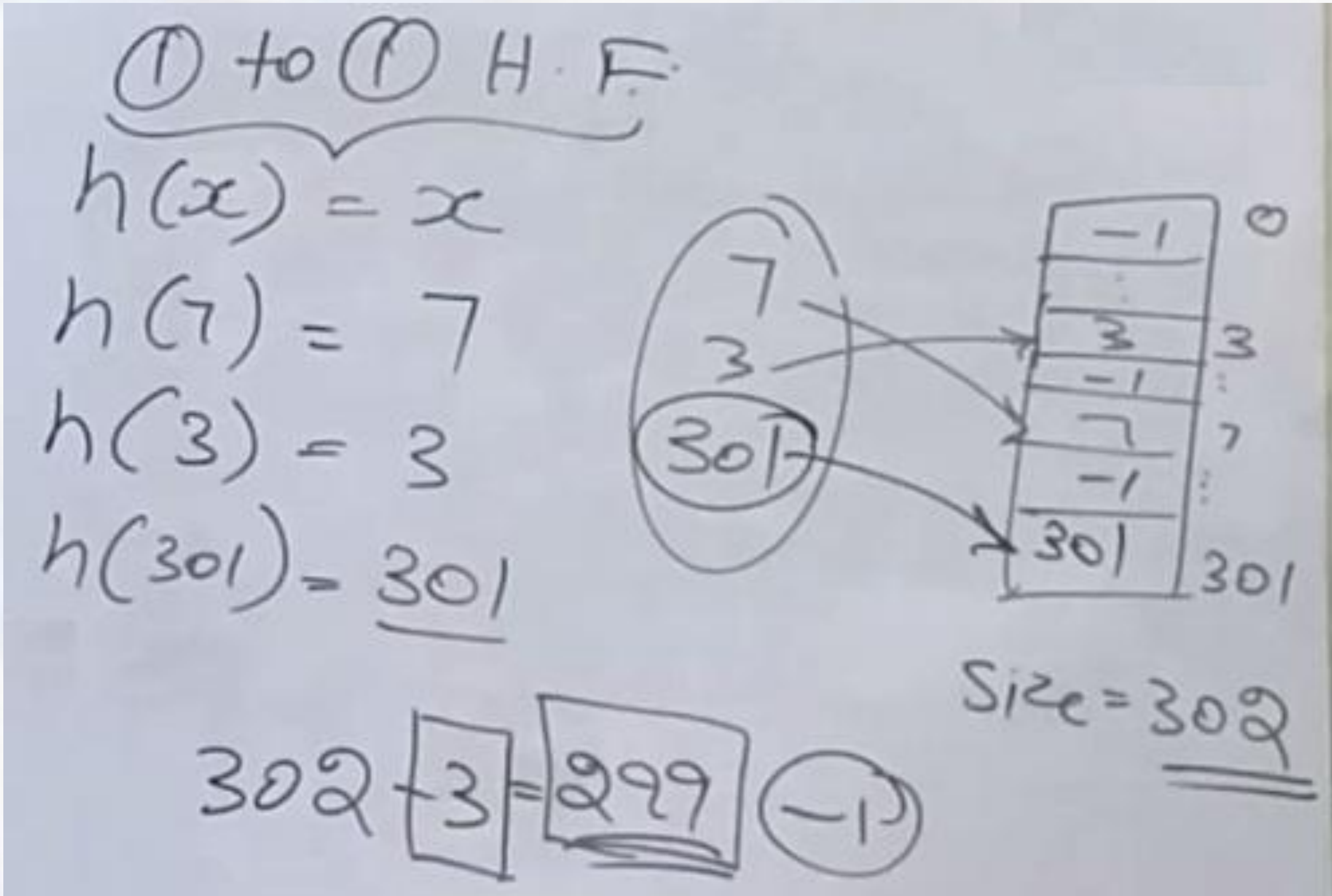
- -1 is a sen

Hashing Function Types

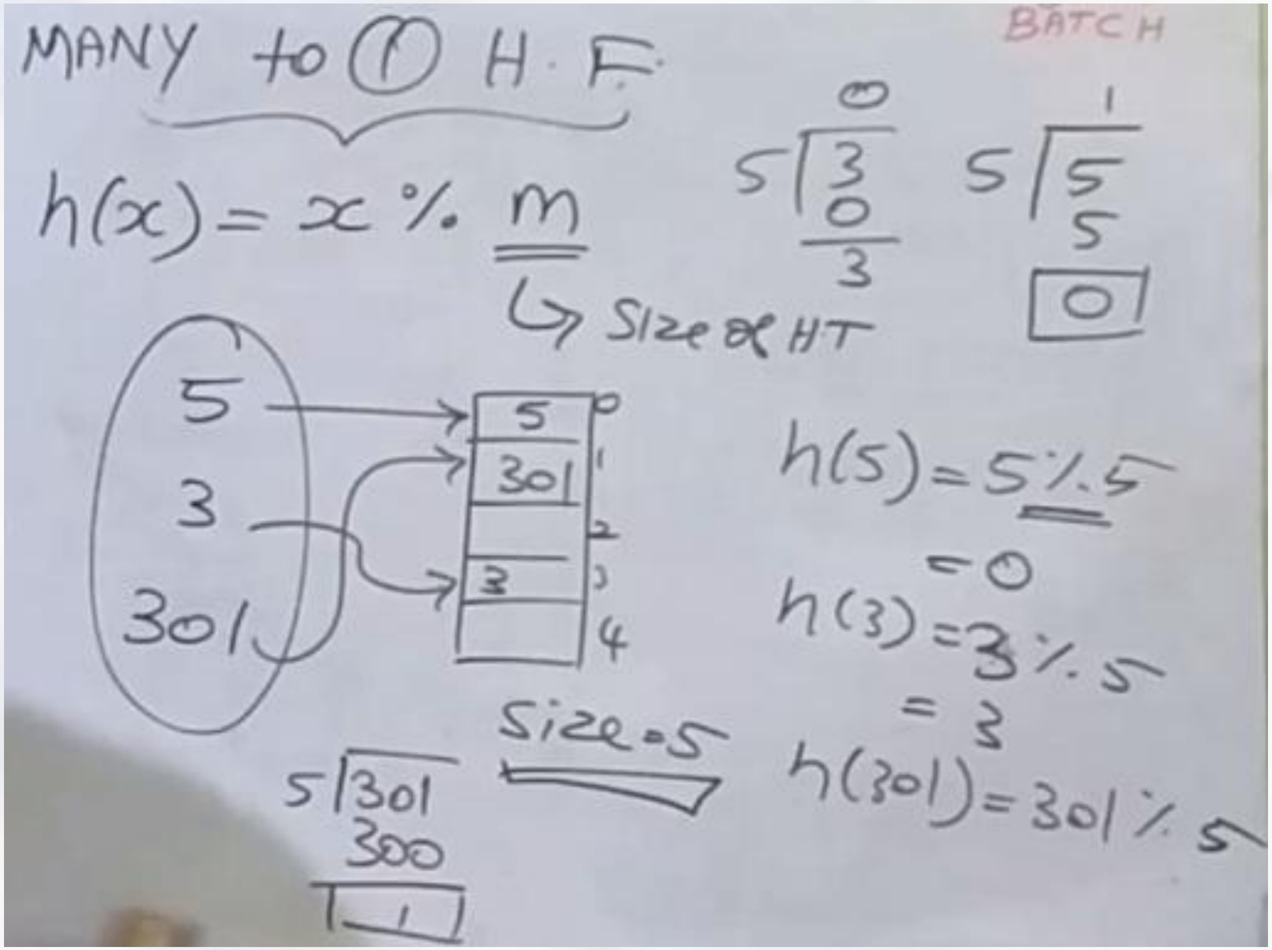
HASH FUNCTION

- **ONE TO ONE**
  - Hash Function : h(x) = x
- **MANY TO ONE**
  - Hash Function : h(x) = x % size

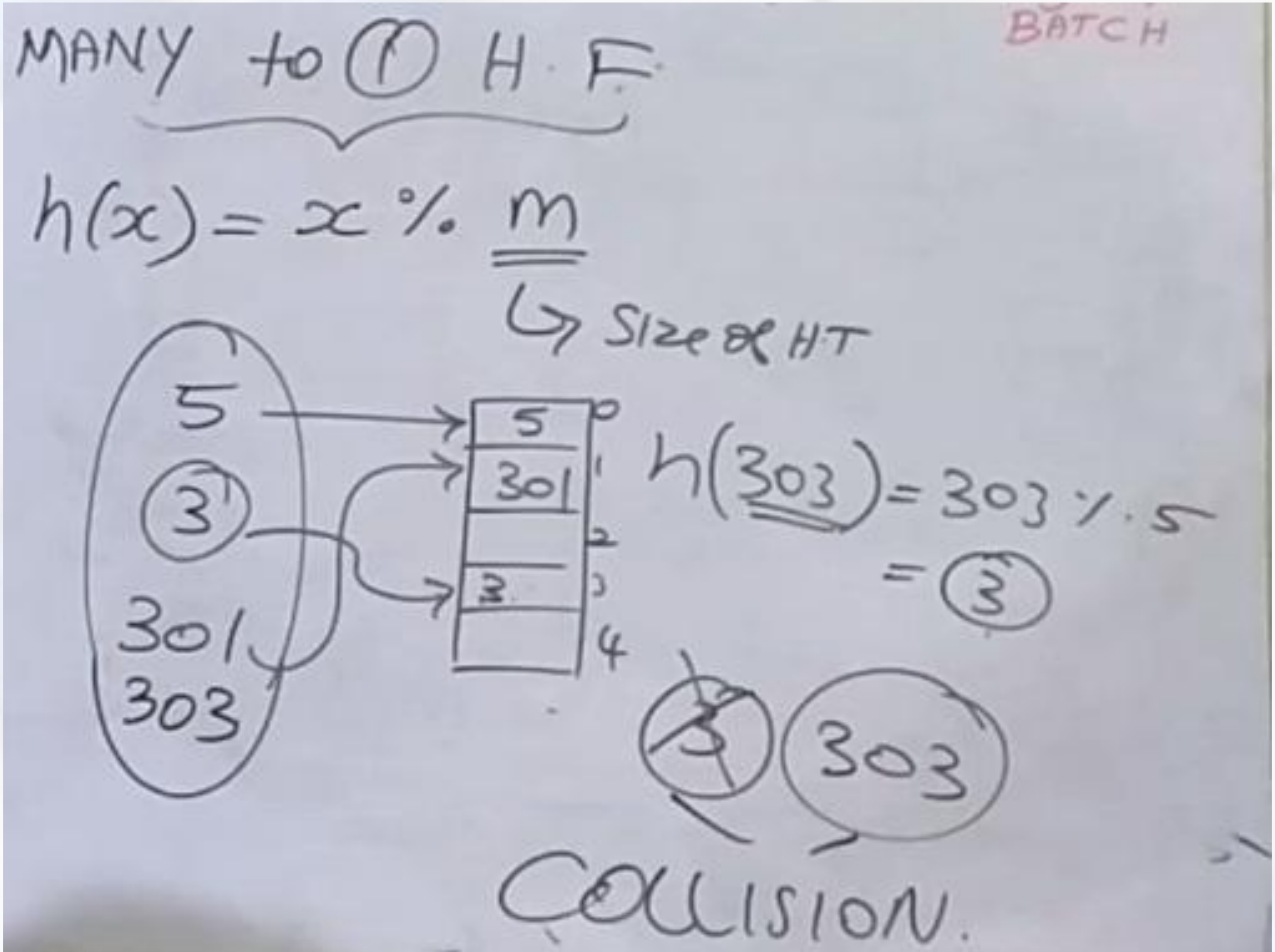
ONE TO ONE Hashing Example



MANY TO ONE Hashing Example



MANY TO ONE Hashing Example



COLLISION RESOLUTION TECHNIQUES

COLLISION RESOLUTION TECHNIQUES

- **OPEN HASHING or CLOSED ADDRESSING**
  - Separate Chaining
- **CLOSED HASHING or OPEN ADDRESSING**
  1) Linear Probing
  2) Quadratic Probing

$h(x) = x \ \% \ size$

$h(x) = x \ \% \ size$ (if collision not occurs)

$h'(x)=[h(x) + f(i)] \ \% \ size$

where

$f(i) = i$

$f(i) = 0, 1, 2, 3, \ldots\ldots\ldots(size-1)$

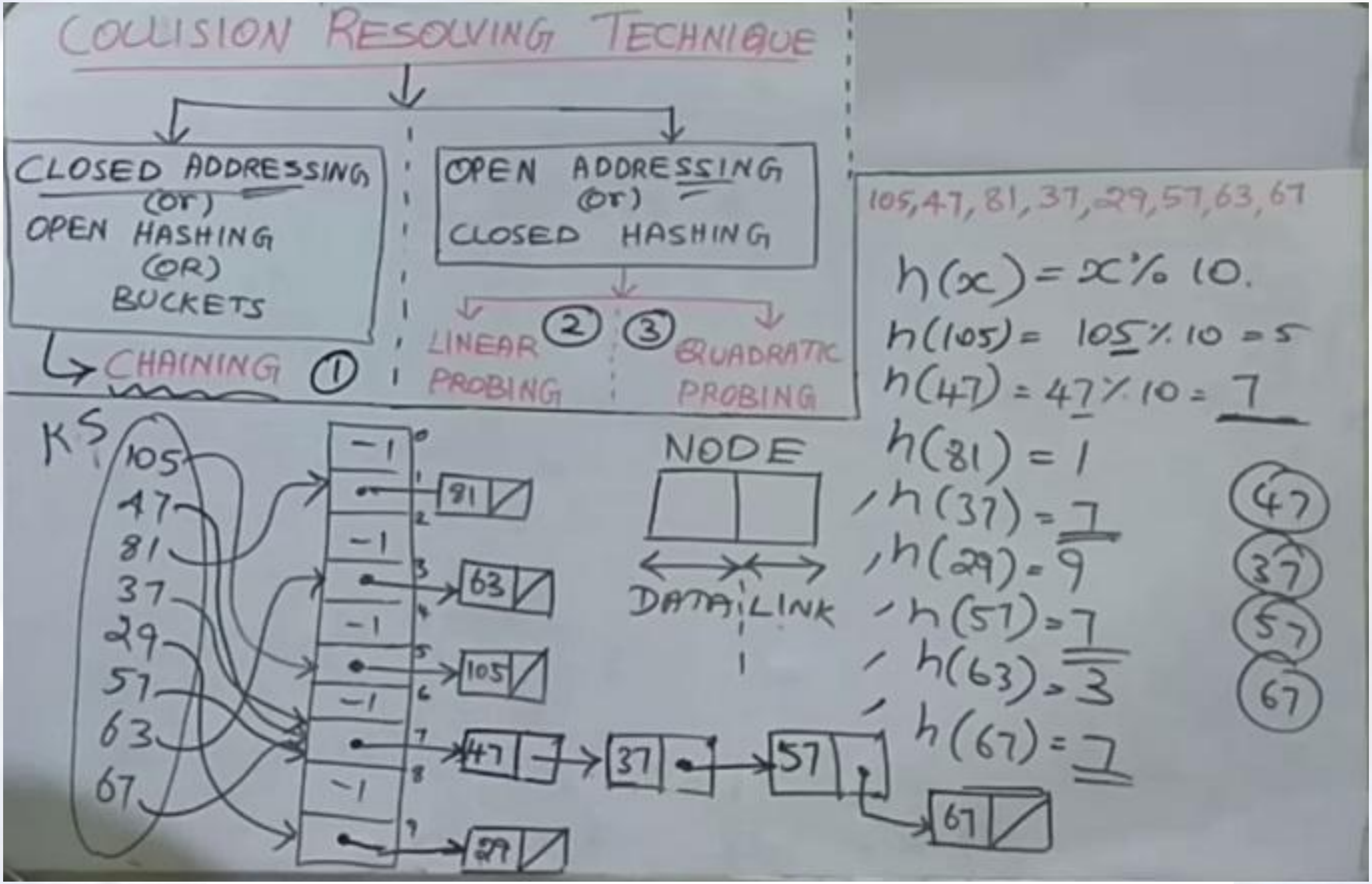
For $i= 0, 1, 2, 3, \ldots\ldots\ldots(size-1)$

$f(i) = i^2$

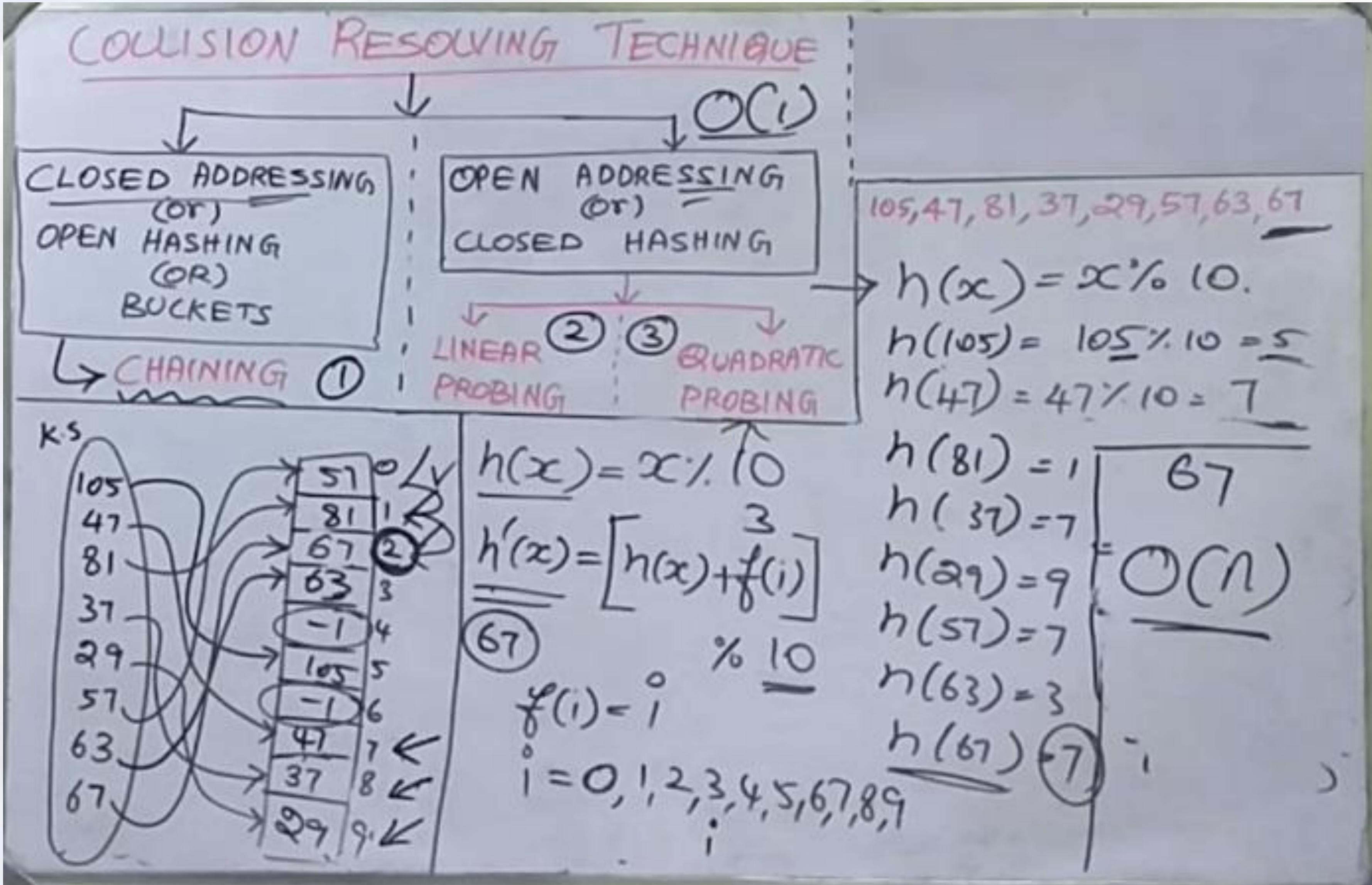
$f(i) = 0, 1, 4, 9, \ldots\ldots\ldots(size-1)^2$

For $i= 0, 1, 2, 3, \ldots\ldots\ldots(size-1)$

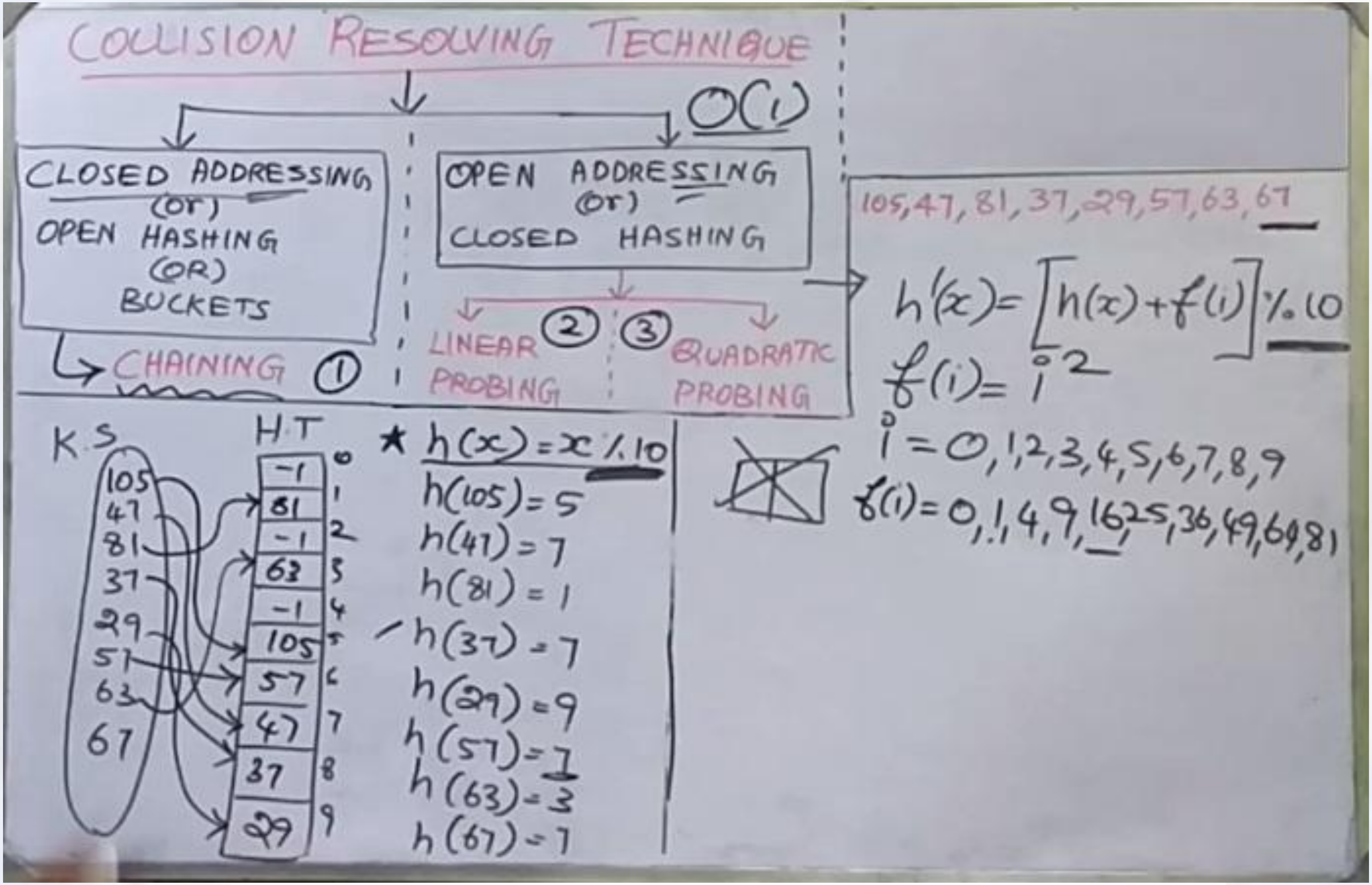
Separate Chaining Example



Linear Probing Example

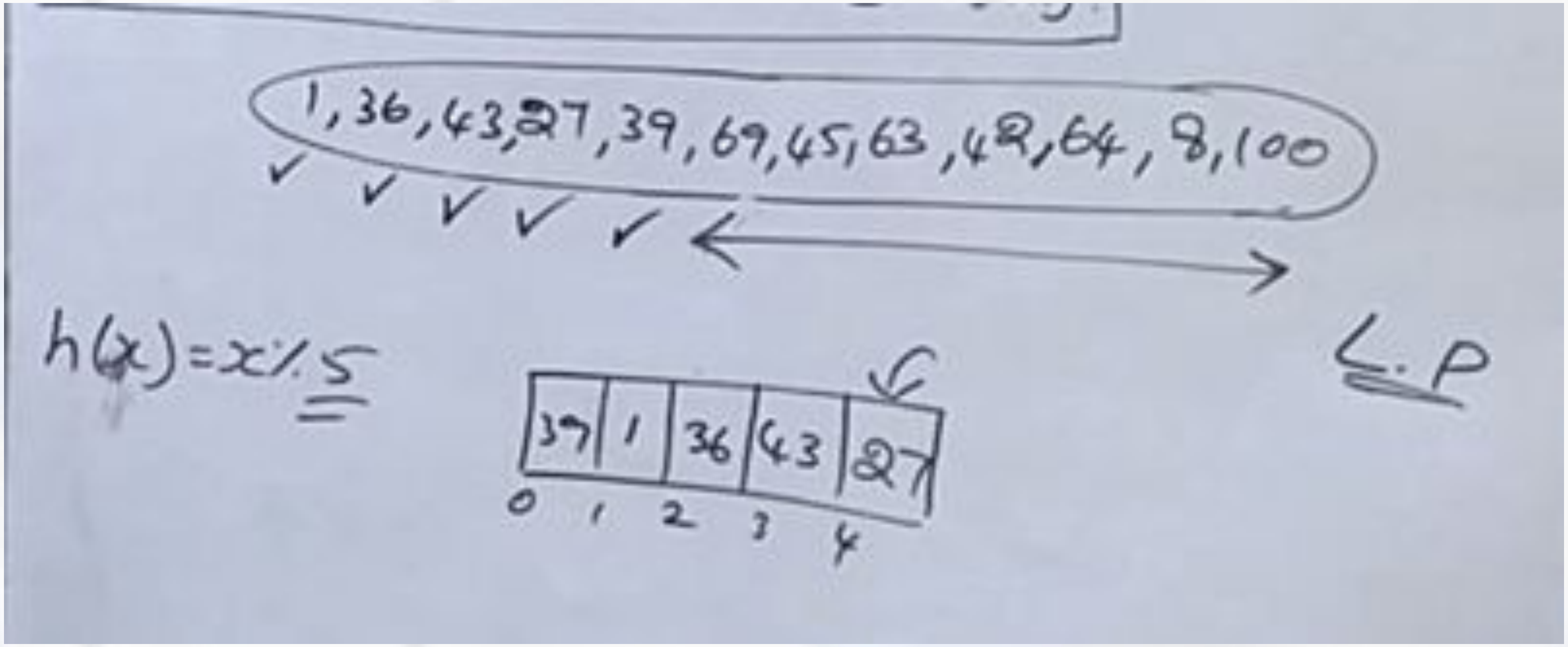


Quadratic Probing Example

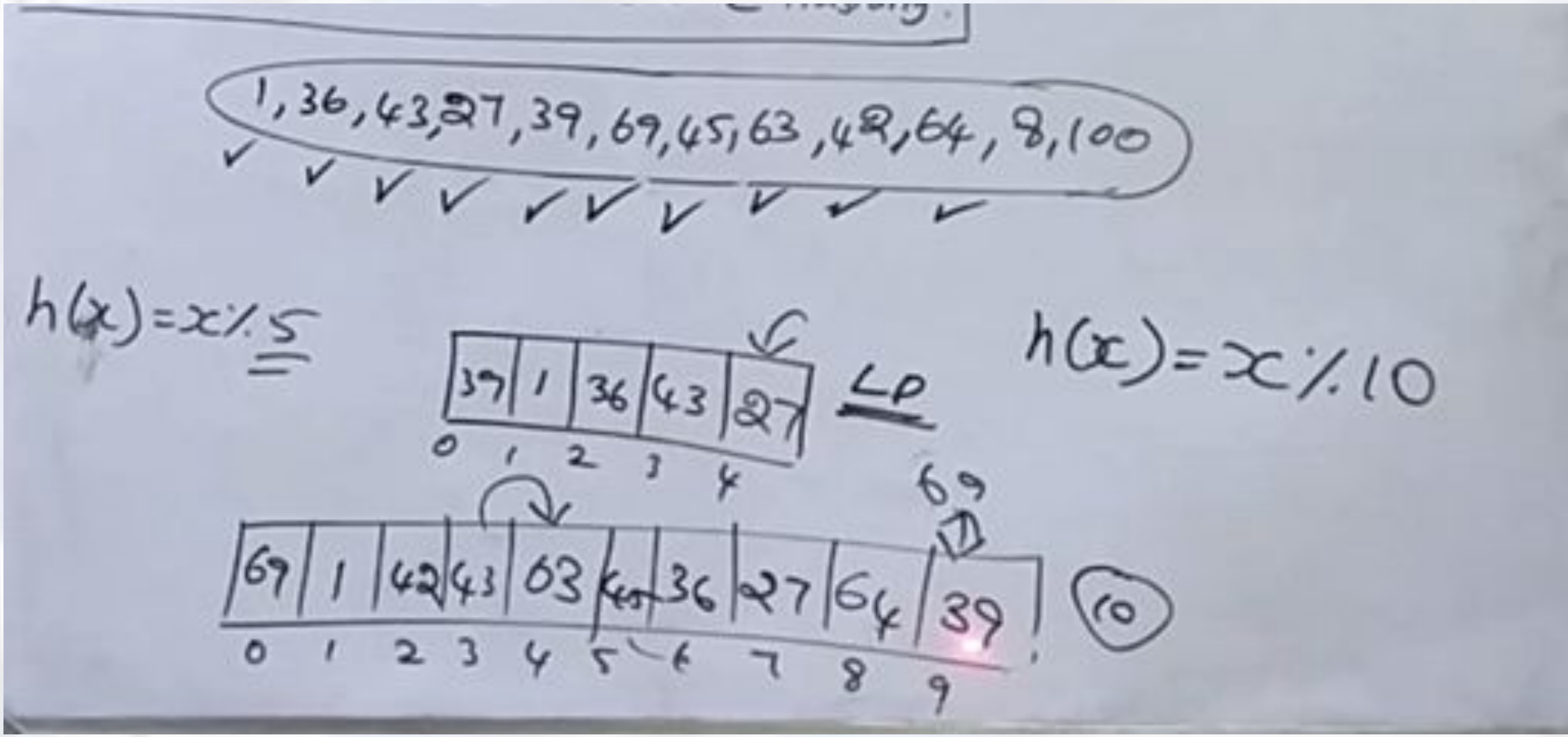


Rehashing

- When hash table is full, the number of collision increases, it degrade the performance.
- It is better to go with new hash table with size double the size of original table. All keys are removed from original hash table.



Rehashing


1, 36, 43, 27, 39, 69, 45, 63, 42, 64, 8, 100
h(x)=x%5
h(x)=x%10
39 | 1 | 36 | 43 | 27
0 1 2 3 4
69 | 1 | 42 | 43 | 63 | 45 | 36 | 27 | 64 | 39
0 1 2 3 4 5 6 7 8 9
69
10

Rehashing Example