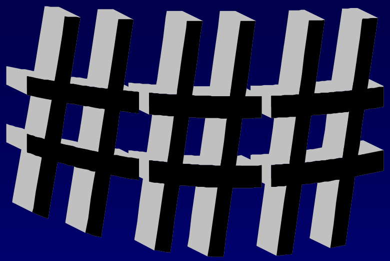
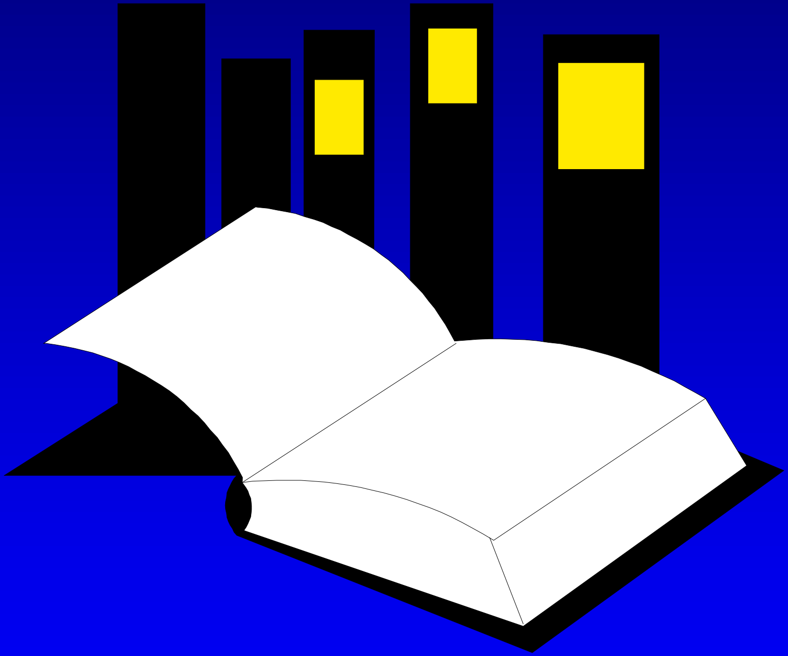
Hash Tables

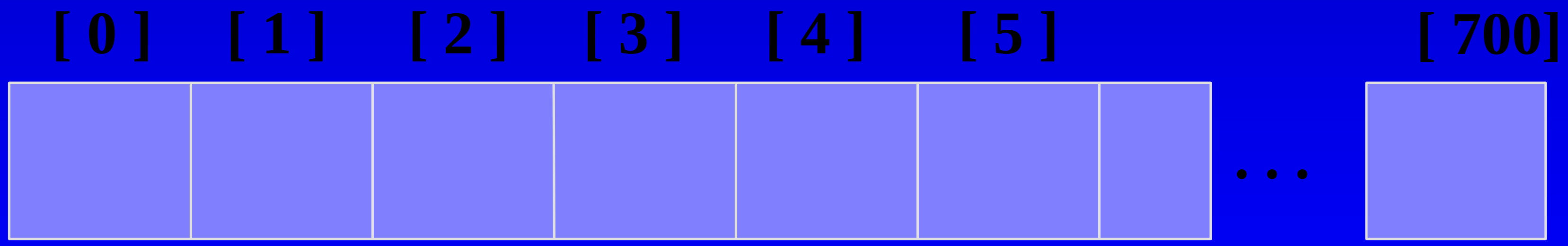




- Hash tables are a common approach to the storing/searching problem.
- This presentation introduces hash tables.

What is a Hash Table ?

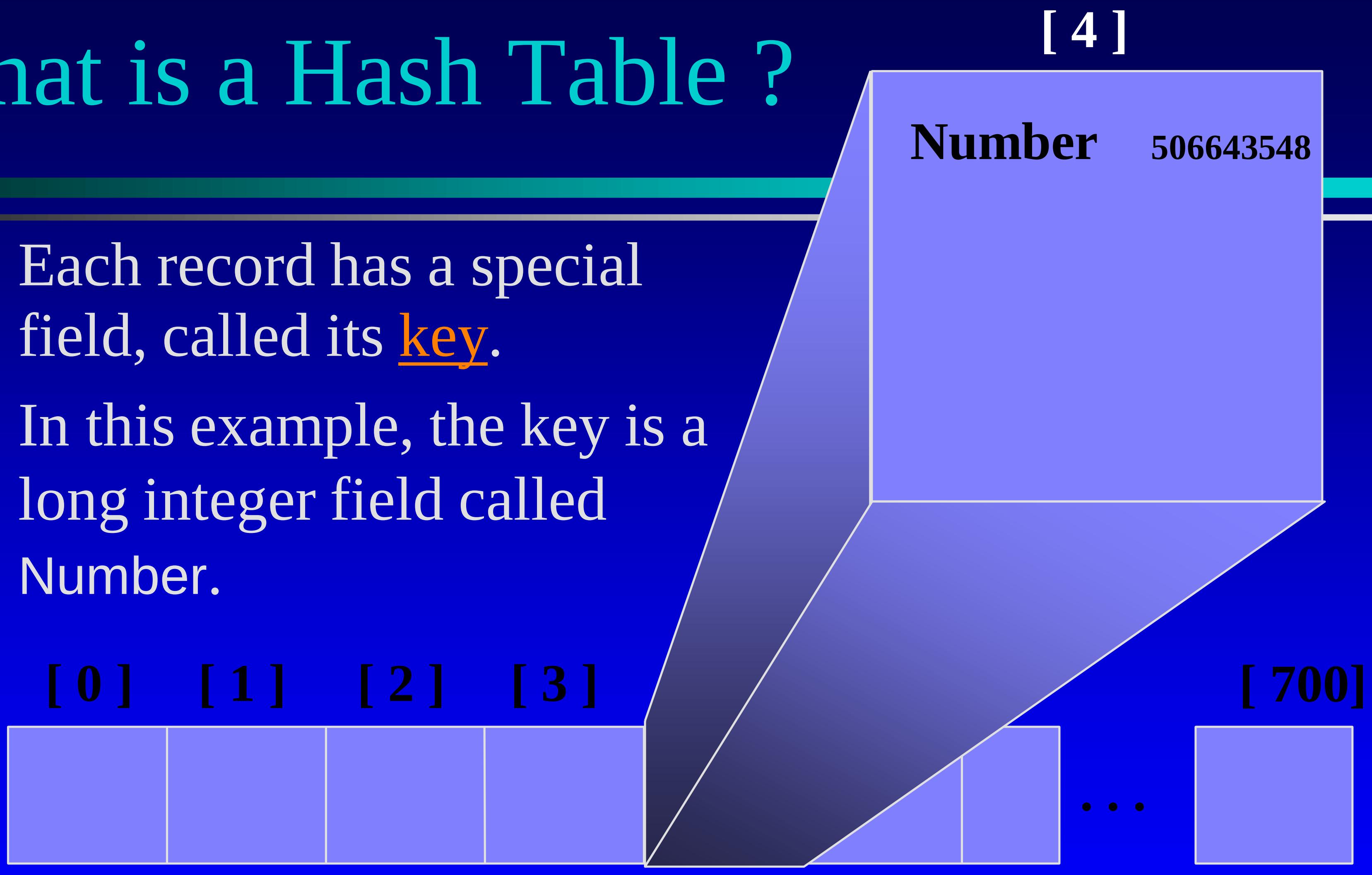
- The simplest kind of hash table is an array of records.
- This example has 701 records.



An array of records

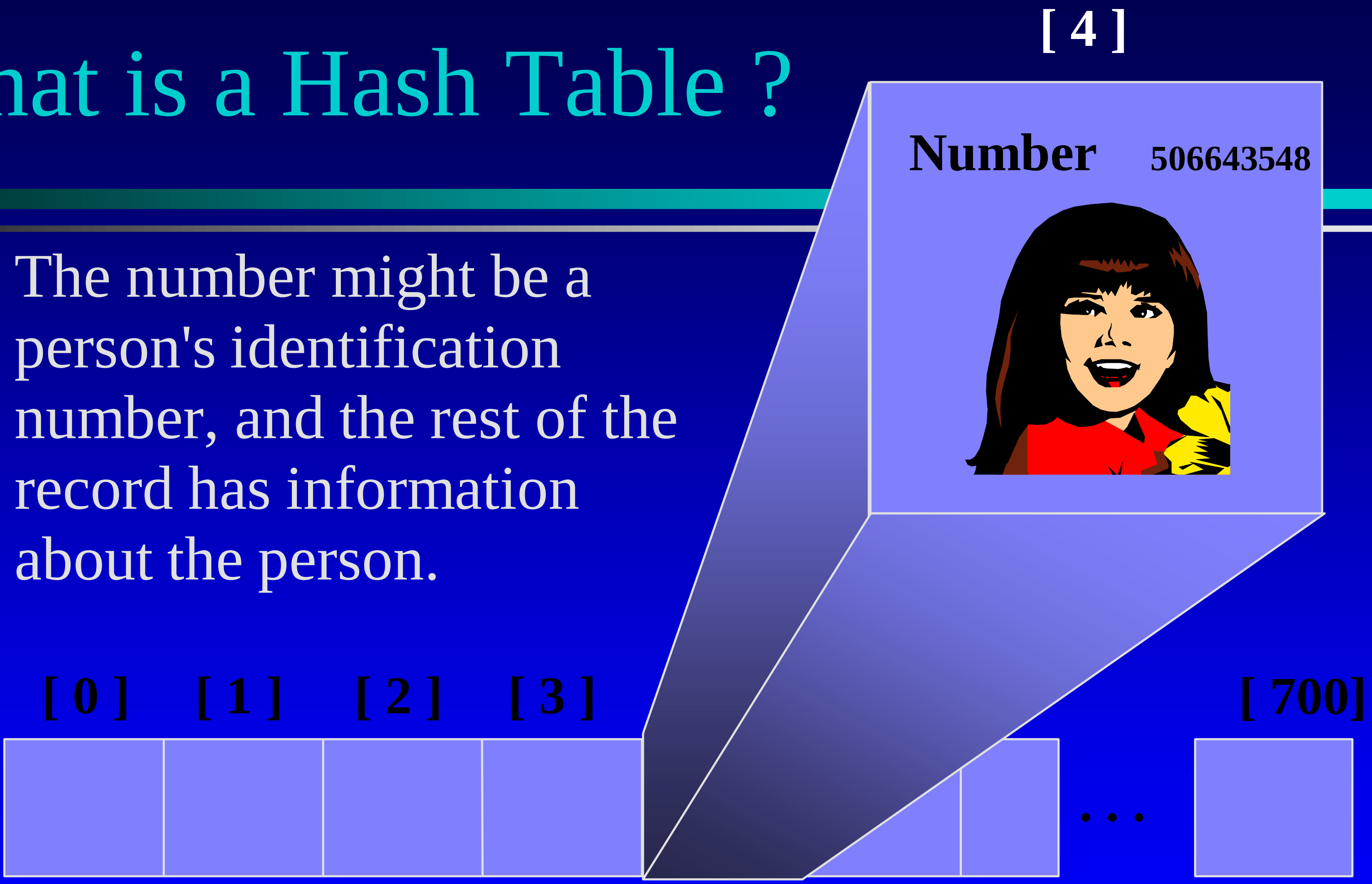
What is a Hash Table ?

- Each record has a special field, called its key.
- In this example, the key is a long integer field called Number.



What is a Hash Table ?

- The number might be a person's identification number, and the rest of the record has information about the person.



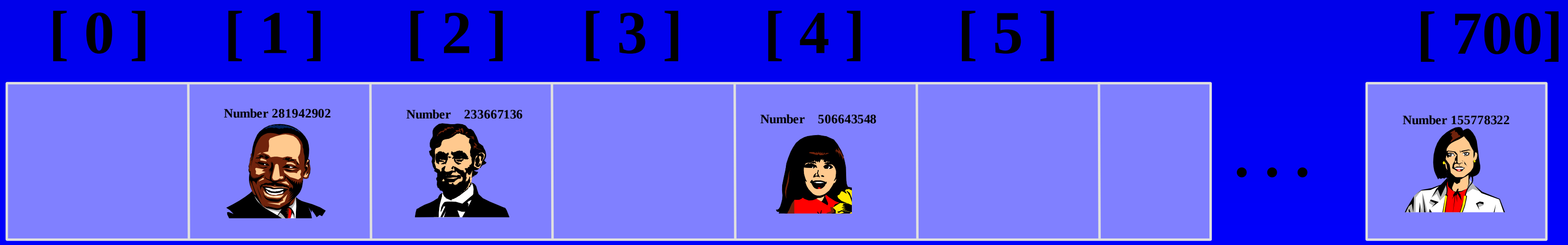
[4]

Number 506643548

[0] [1] [2] [3] . . . [700]

What is a Hash Table ?

- When a hash table is in use, some spots contain valid records, and other spots are "empty".



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

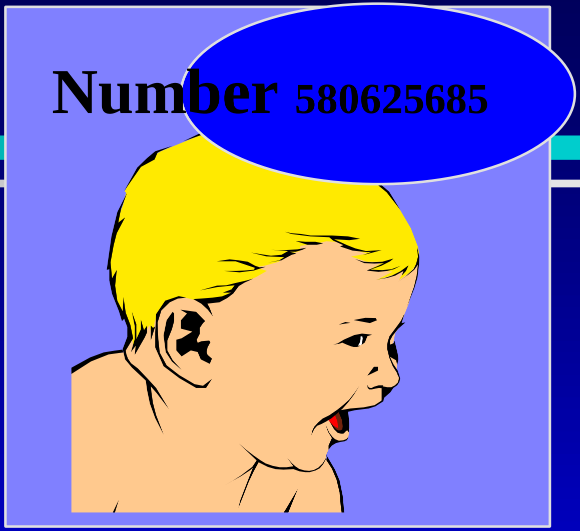
Number 233667136

Number 506643548

. . .

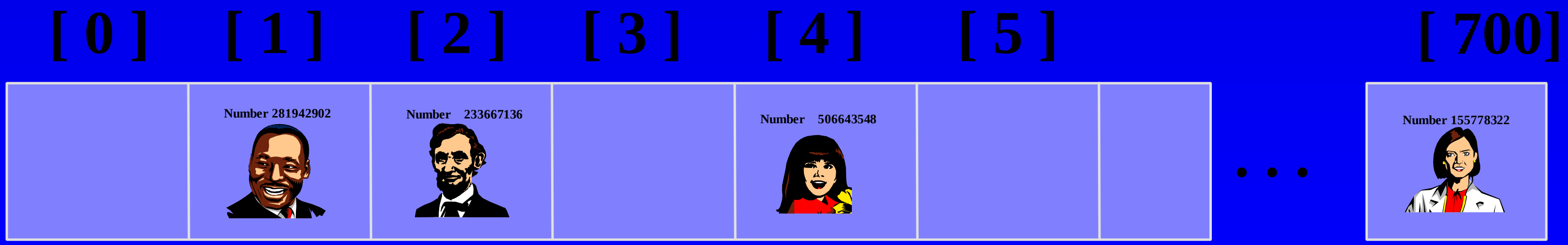
Number 155778322

Inserting a New Record



Number 580625685

- In order to insert a new record, the **key** must somehow be **converted to** an array **index**.
- The index is called the **hash value** of the key.



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 506643548

. . .

Number 155778322

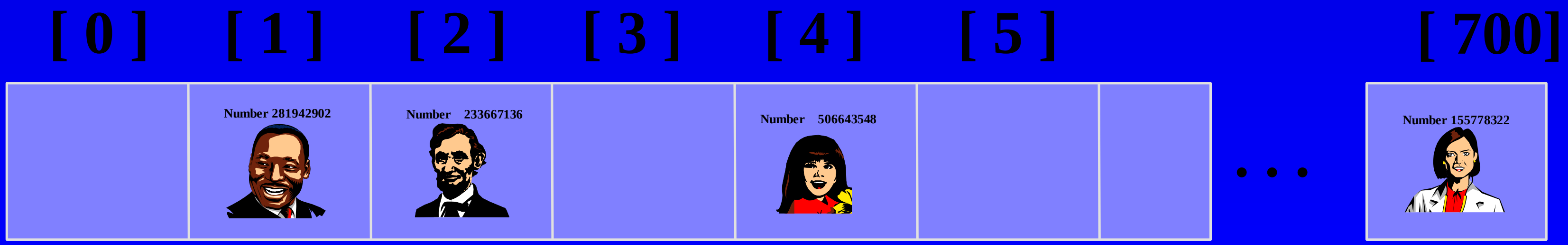
Inserting a New Record

Number 580625685

- Typical way create a hash value:

  (Number mod 701)

What is (580625685 mod 701) ?



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 506643548

. . .

Number 155778322

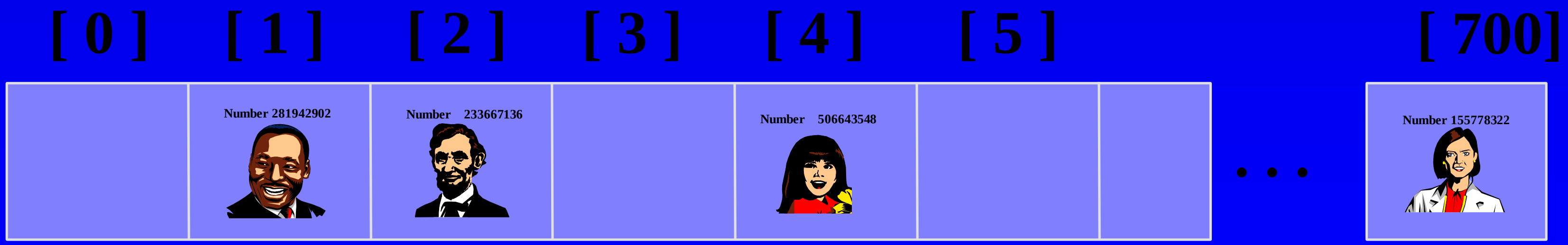
Inserting a New Record

- Typical way to create a hash value:

(Number mod 701)

What is (580625685 mod 701) ?

Number 580625685
3


[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

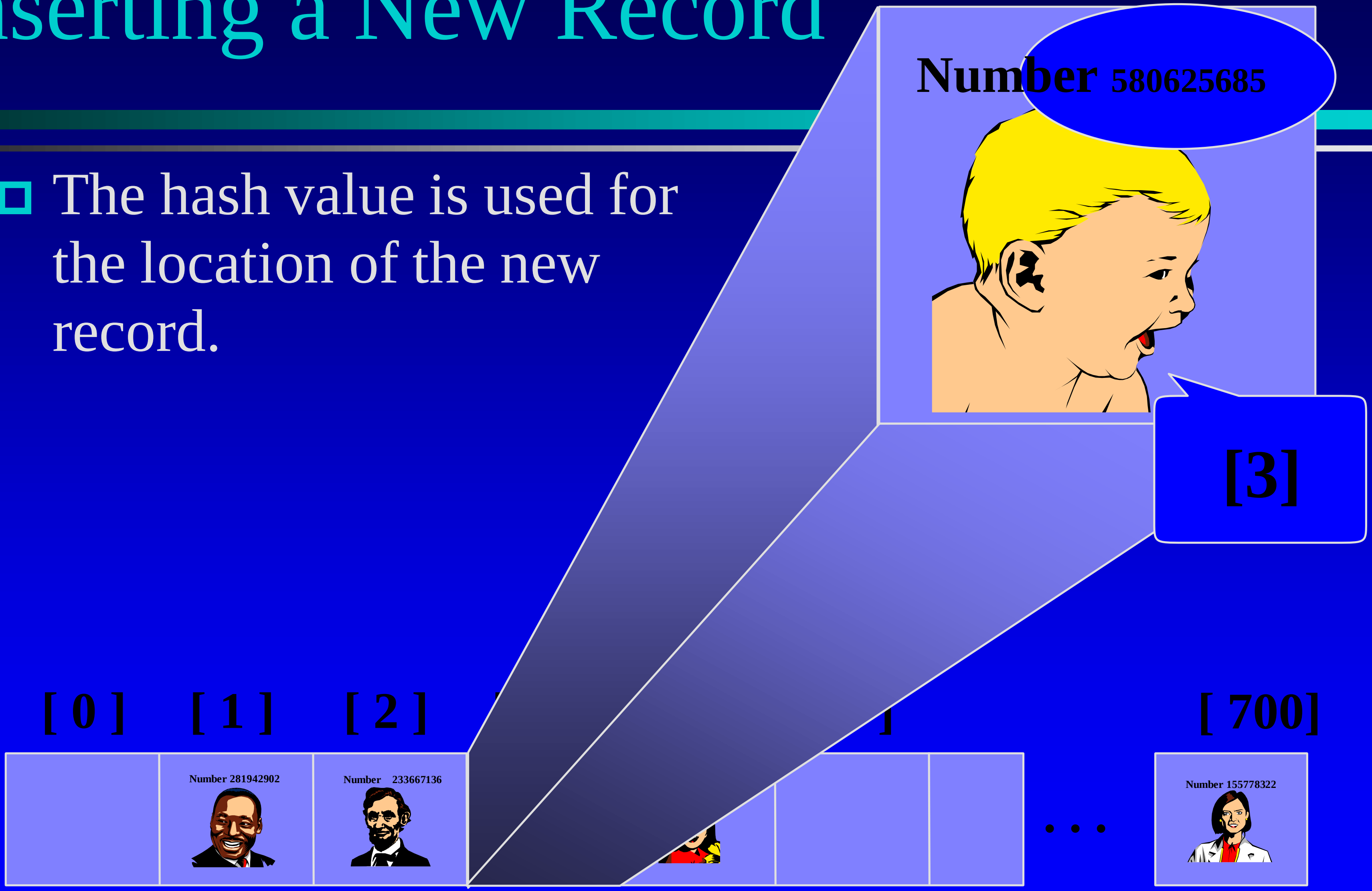
Number 506643548

. . .

Number 155778322

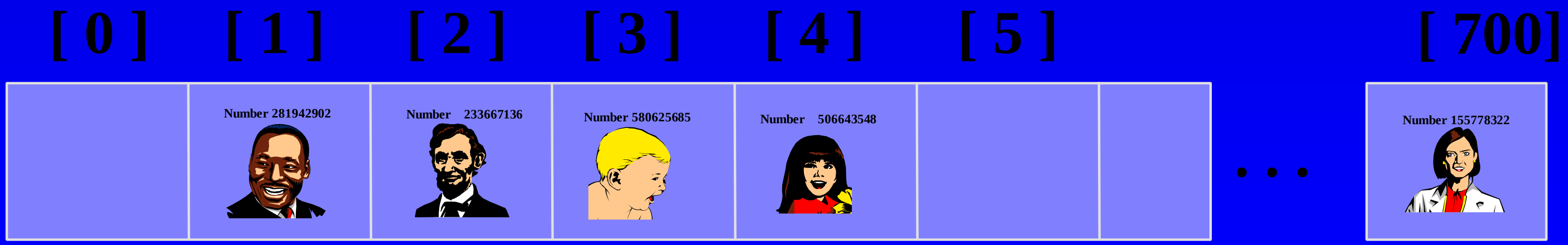
Inserting a New Record

- The hash value is used for the location of the new record.



Inserting a New Record

- The hash value is used for the location of the new record.



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 580625685

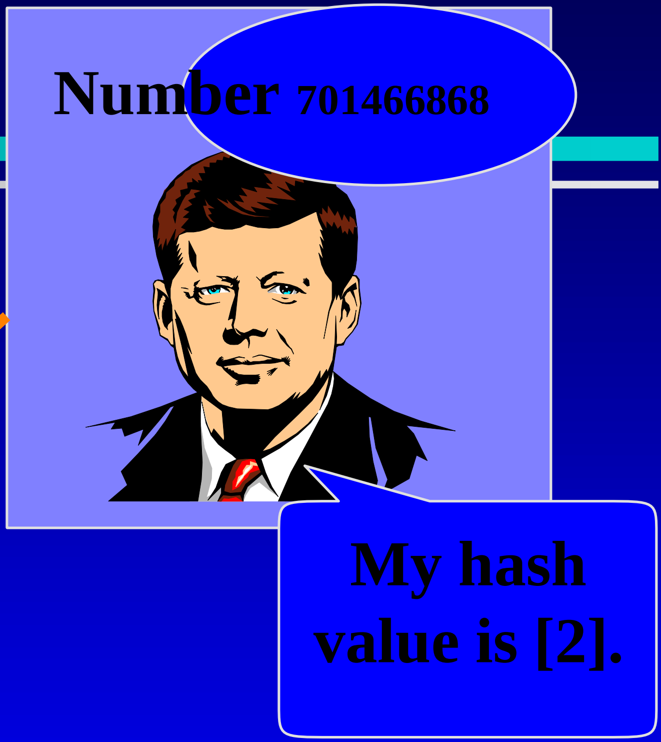
Number 506643548

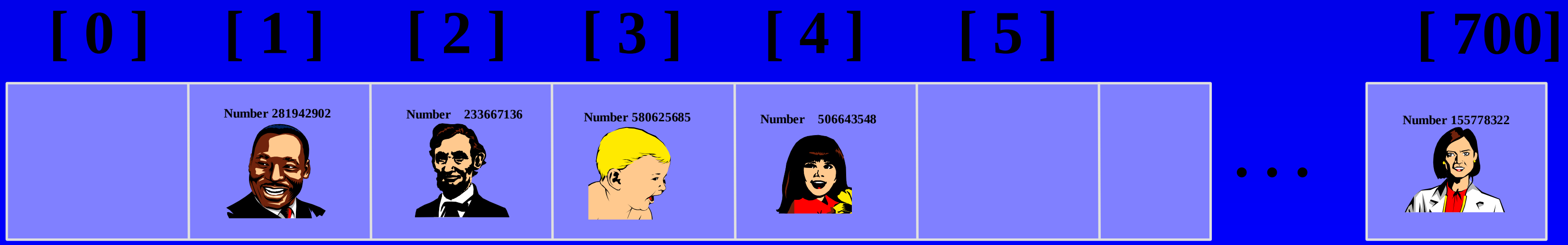
. . .

Number 155778322

Collisions

- Here is another new record to insert, with a hash value of 2.

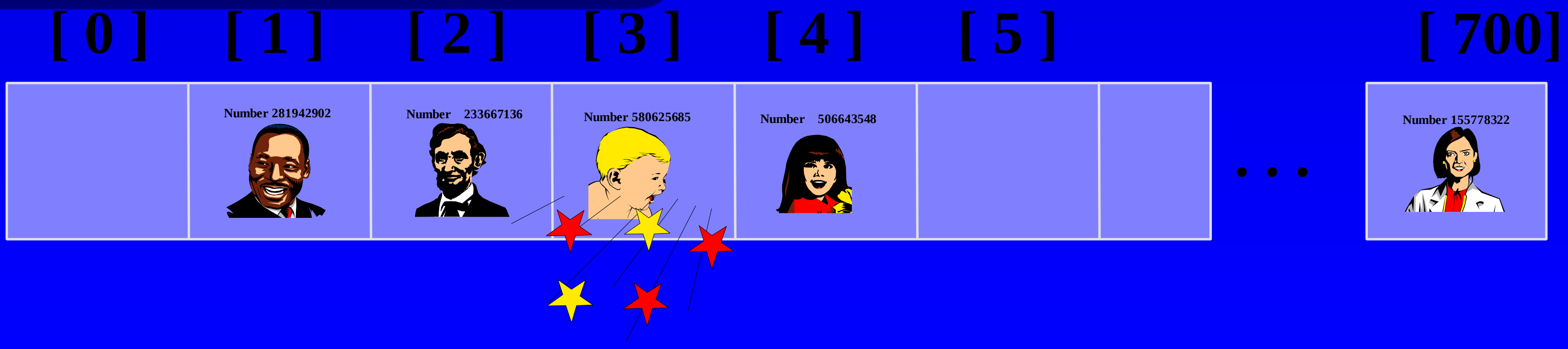




Collisions

- This is called a **collision**, because there is already another valid record at [2].

When a collision occurs, move forward until you find an empty spot.

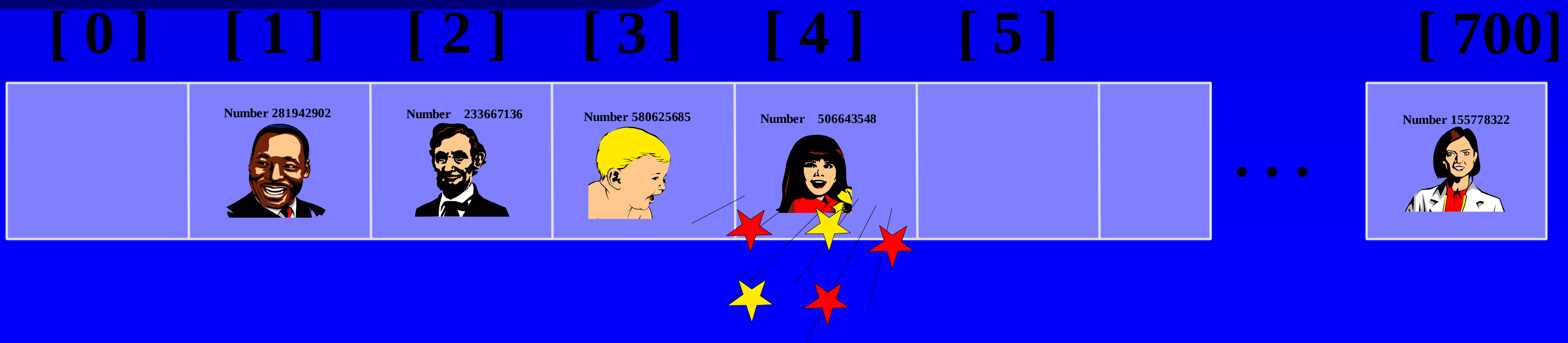


Collisions

- This is called a **collision**, because there is already another valid record at [2].

When a collision occurs,
move forward until you find an empty spot.

Number 701466868

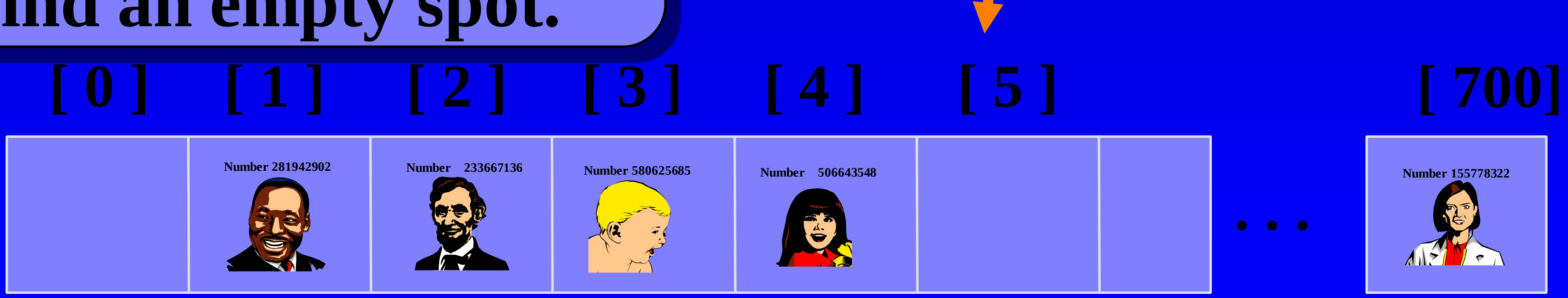


Collisions

- This is called a **collision**, because there is already another valid record at [2].

**When a collision occurs,
move forward until you find an empty spot.**

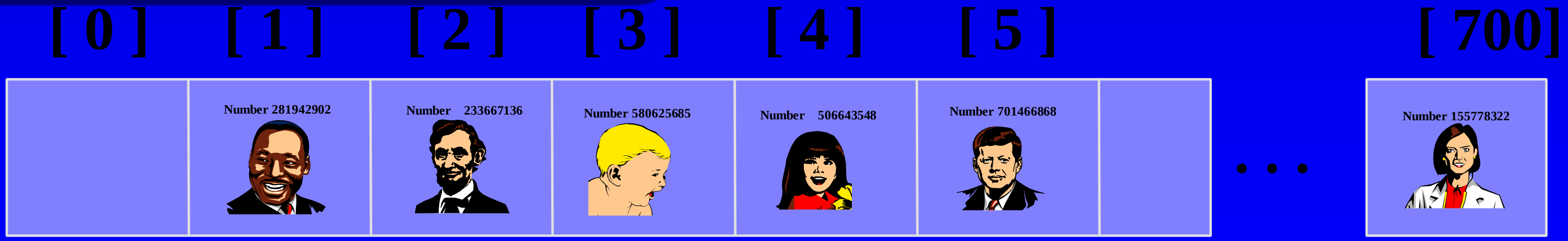
Number 701466868



Collisions

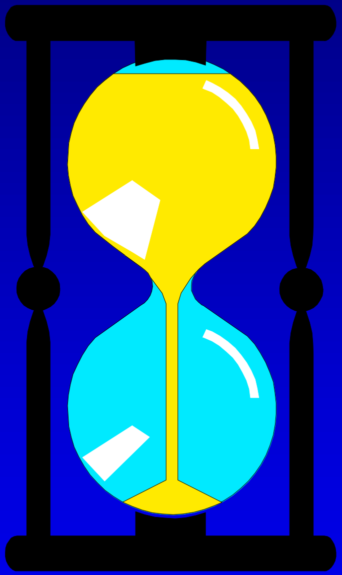
- This is called a **collision**, because there is already another valid record at [2].

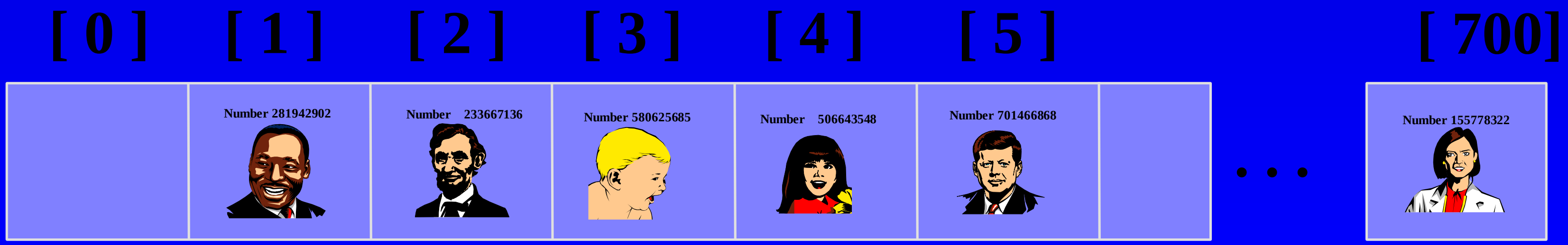
The new record goes in the empty spot.



A Quiz

Where would you be placed in this table, if there is no collision? Use your social security number or some other favorite number.





[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 580625685

Number 506643548

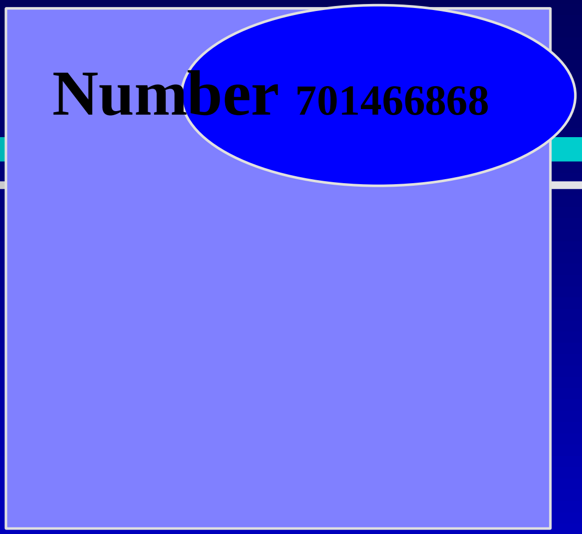
Number 701466868

. . .

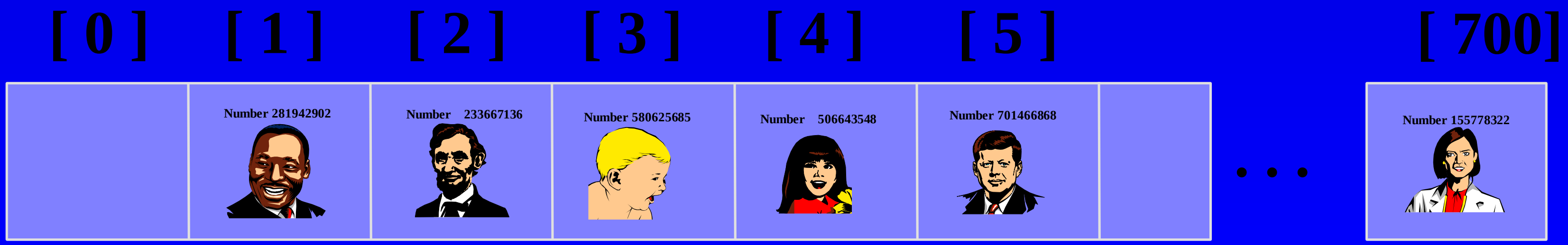
Number 155778322

Searching for a Key

- The data that's attached to a key can be found fairly quickly.



Number 701466868



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 580625685

Number 506643548

Number 701466868

. . .

Number 155778322

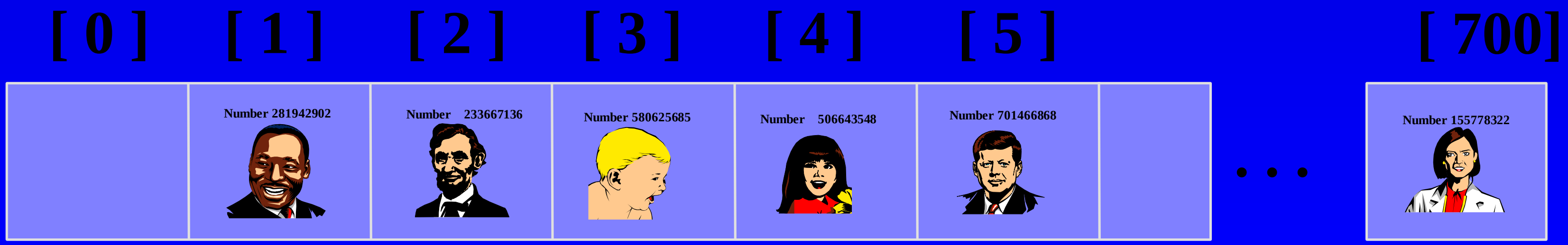
Searching for a Key

- Calculate the hash value.
- Check that location of the array for the key.

Number 701466868

My hash value is [2].

Not me.



[0]	[1]	[2]	[3]	[4]	[5]			[700]
	Number 281942902	Number 233667136	Number 580625685	Number 506643548	Number 701466868		. . .	Number 155778322

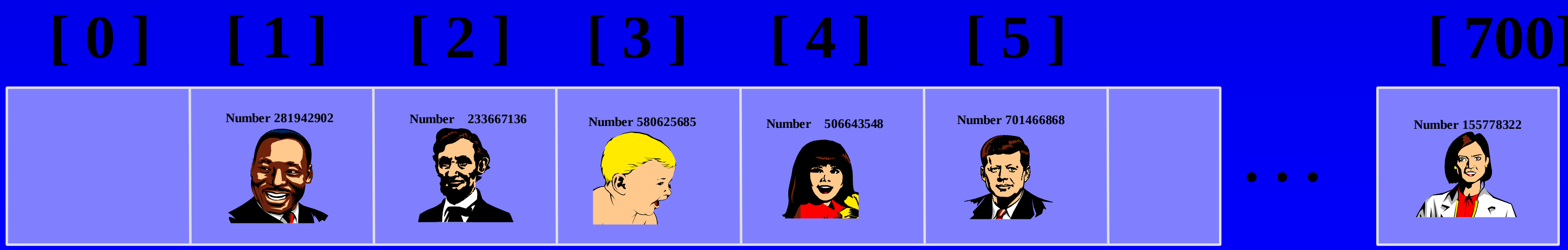
Searching for a Key

- Keep moving forward until you find the key, or you reach an empty spot.

Number 701466868

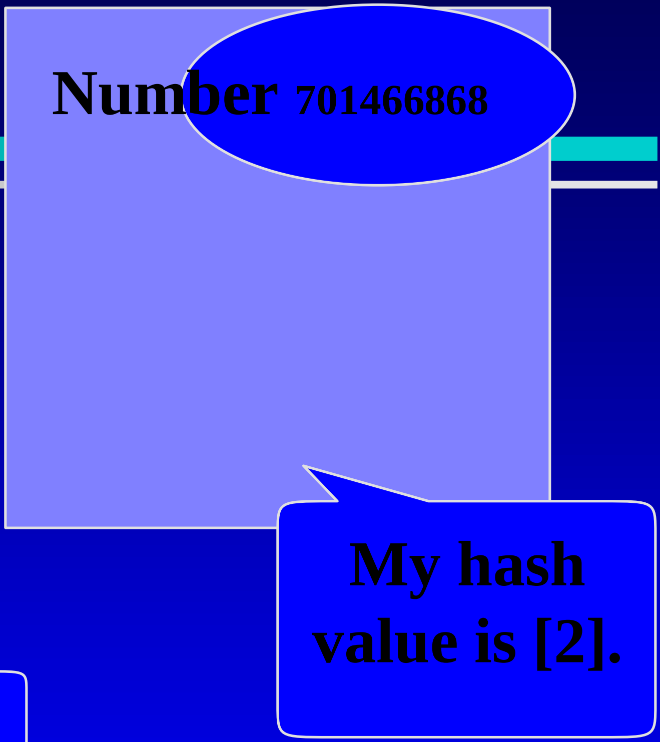
My hash value is [2].

Not me.

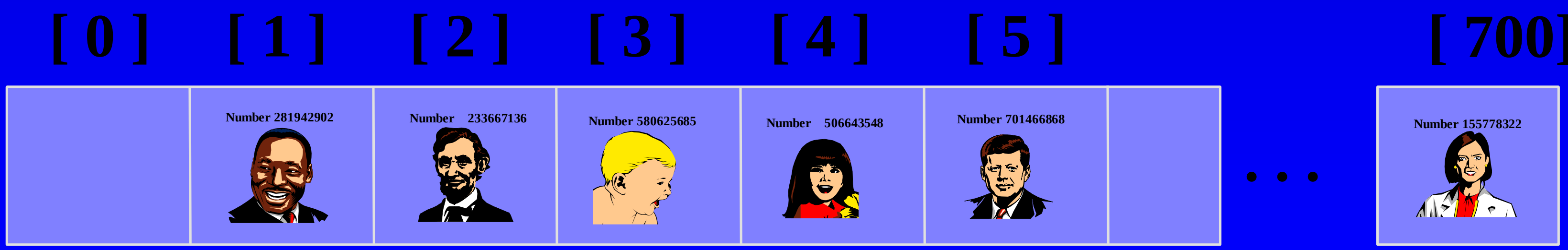


Searching for a Key

- Keep moving forward until you find the key, or you reach an empty spot.

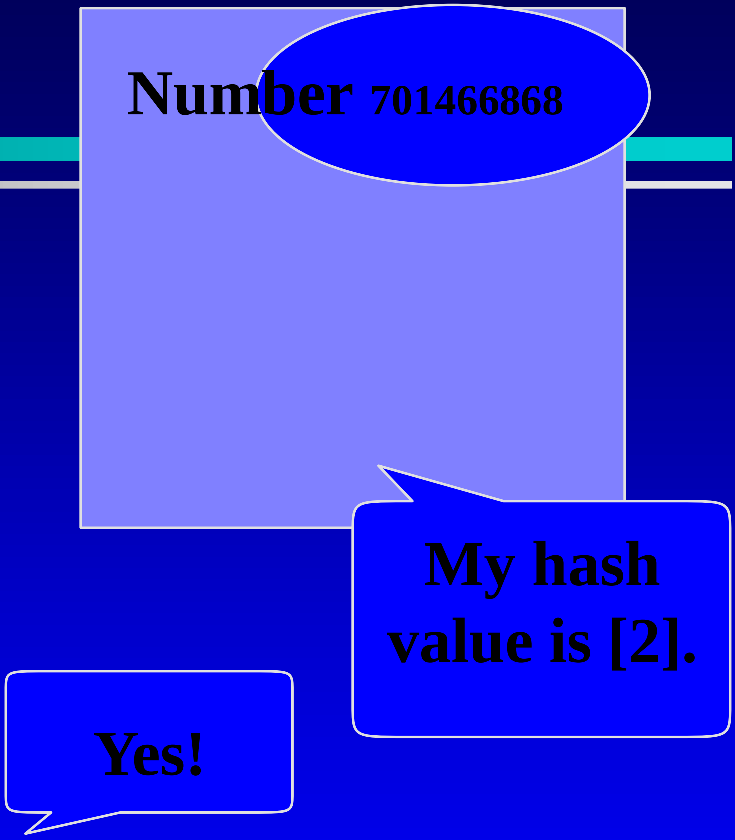


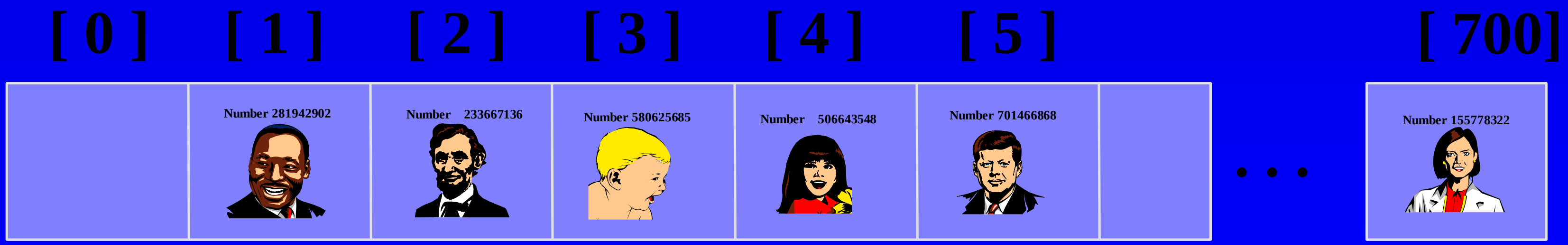
Not me.



Searching for a Key

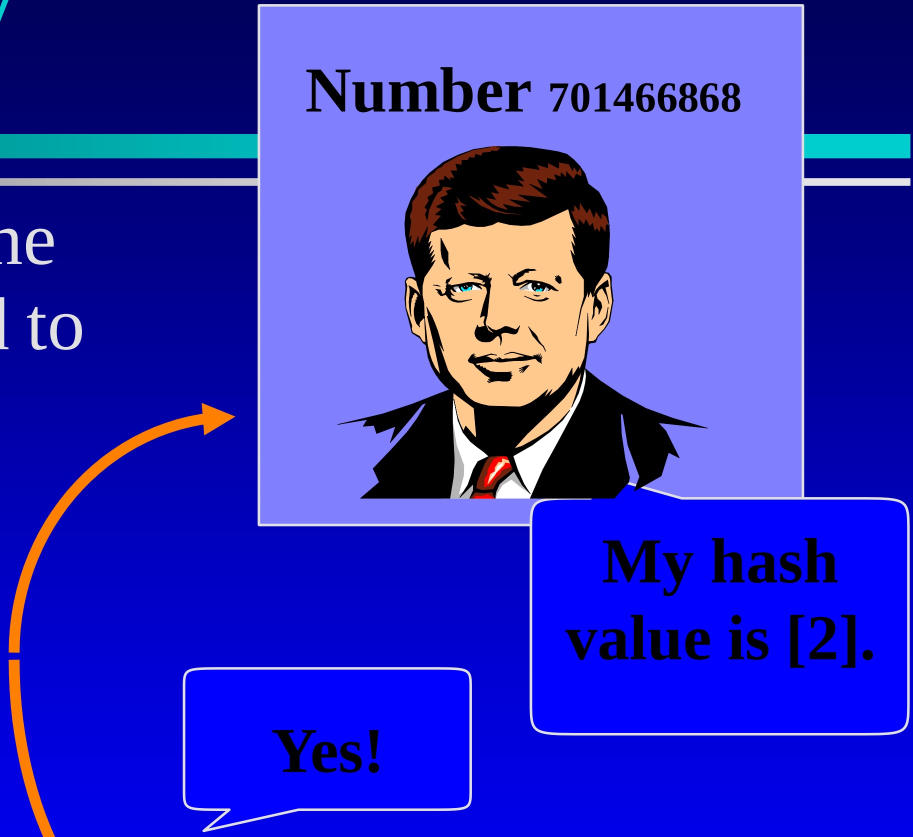
- Keep moving forward until you find the key, or you reach an empty spot.

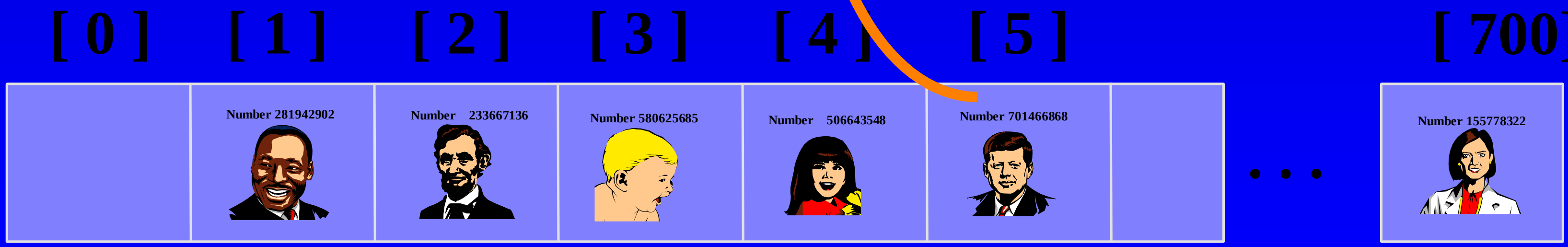




Searching for a Key

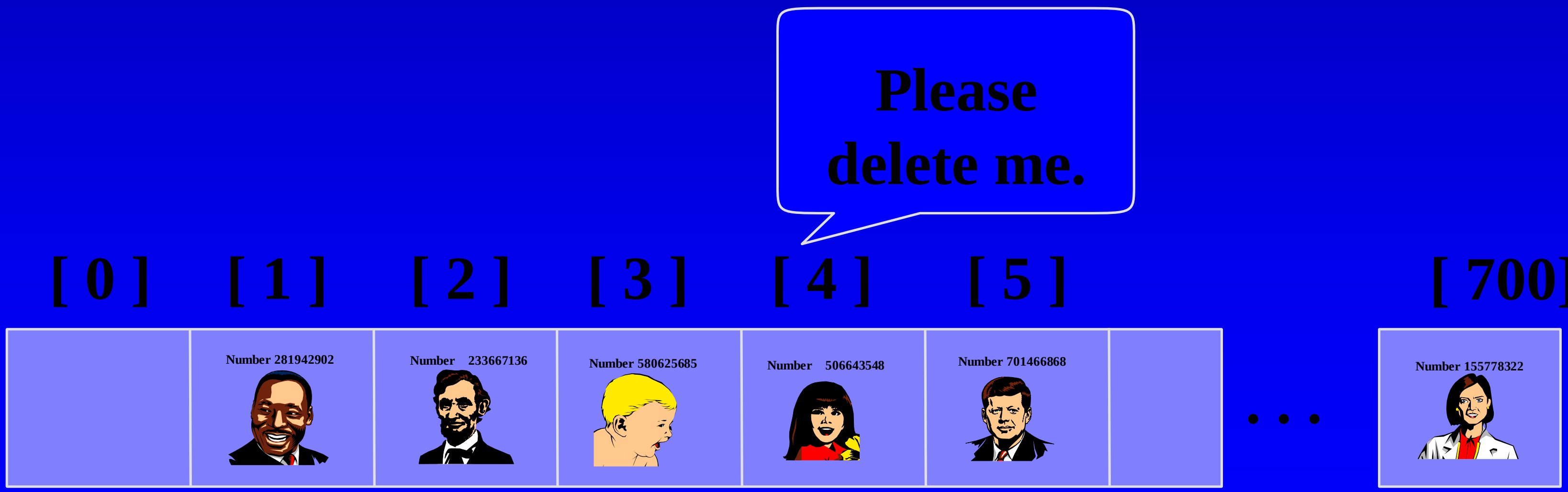
- When the item is found, the information can be copied to the necessary location.





Deleting a Record

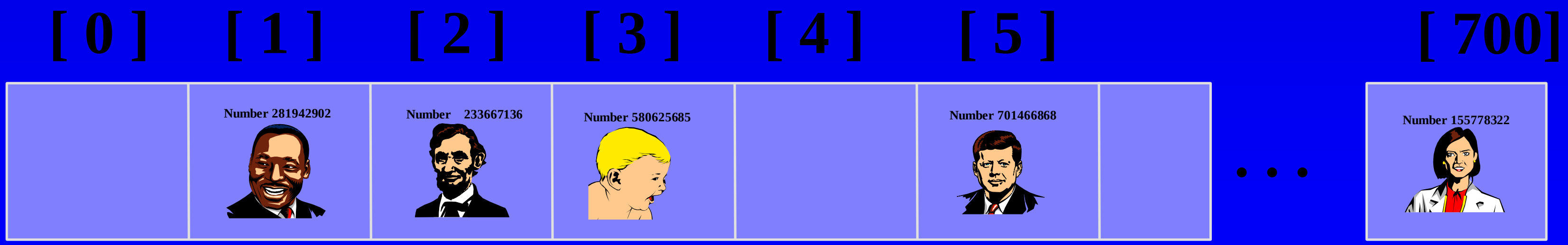
- Records may also be deleted from a hash table.



[0]	[1]	[2]	[3]	[4]	[5]			[700]
	Number 281942902	Number 233667136	Number 580625685	Number 506643548	Number 701466868		. . .	Number 155778322

Deleting a Record

- Records may also be deleted from a hash table.
- But the location must not be left as an ordinary "empty spot" since that could interfere with searches.



[0] [1] [2] [3] [4] [5] [700]

Number 281942902

Number 233667136

Number 580625685

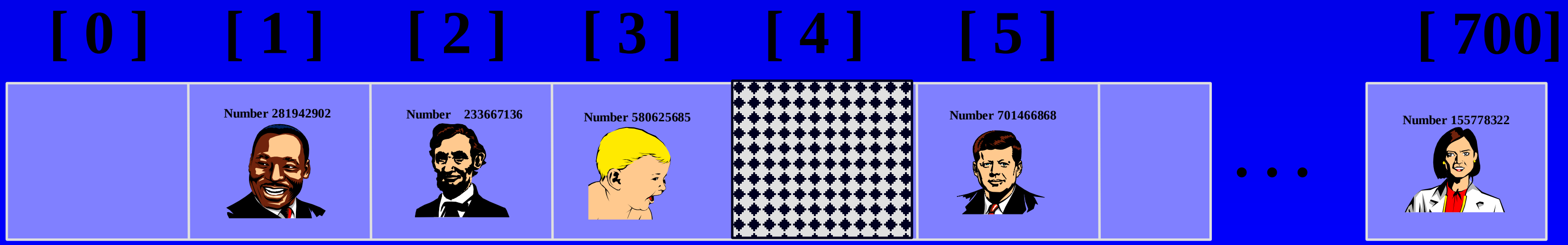
Number 701466868

. . .

Number 155778322

Deleting a Record

- Records may also be deleted from a hash table.
- But the location must not be left as an ordinary "empty spot" since that could interfere with searches.
- The location must be marked in some special way so that a search can tell that the spot used to have something in it.

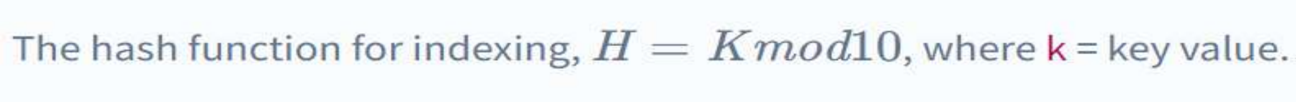


Summary

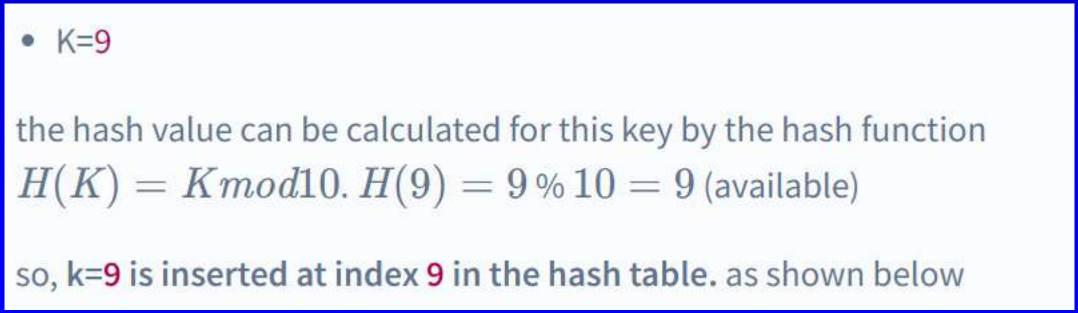
- Hash tables store a collection of records with keys.
- The location of a record depends on the hash value of the record's key.
- When a collision occurs, the next available location is used.
- Searching for a particular key is generally quick.
- When an item is deleted, the location must be marked in a special way, so that the searches know that the spot used to be used.

Quadratic probing

- Let there a table of **size** = 10 with slot position **index** i=0, 1, 2, 3, 4, 5 ,6 ,7,8,9



The hash function for indexing, $H = Kmod10$, where k = key value.

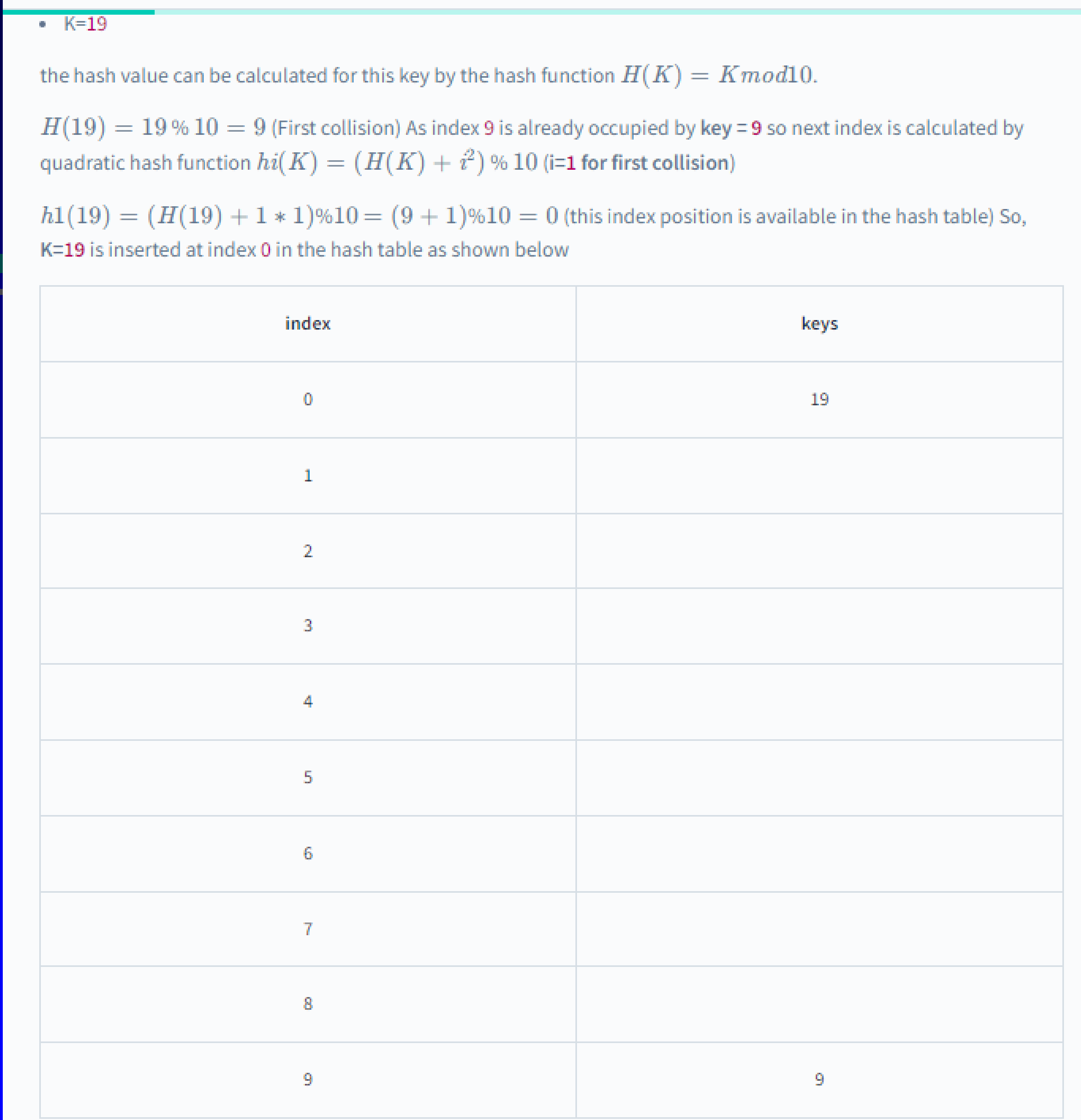


- K=9

the hash value can be calculated for this key by the hash function $H(K) = Kmod10. H(9) = 9 \% 10 = 9$ (available)

so, **k=9 is inserted at index 9 in the hash table.** as shown below

1	
2	
3	
4	
5	
6	
7	
8	
9	9



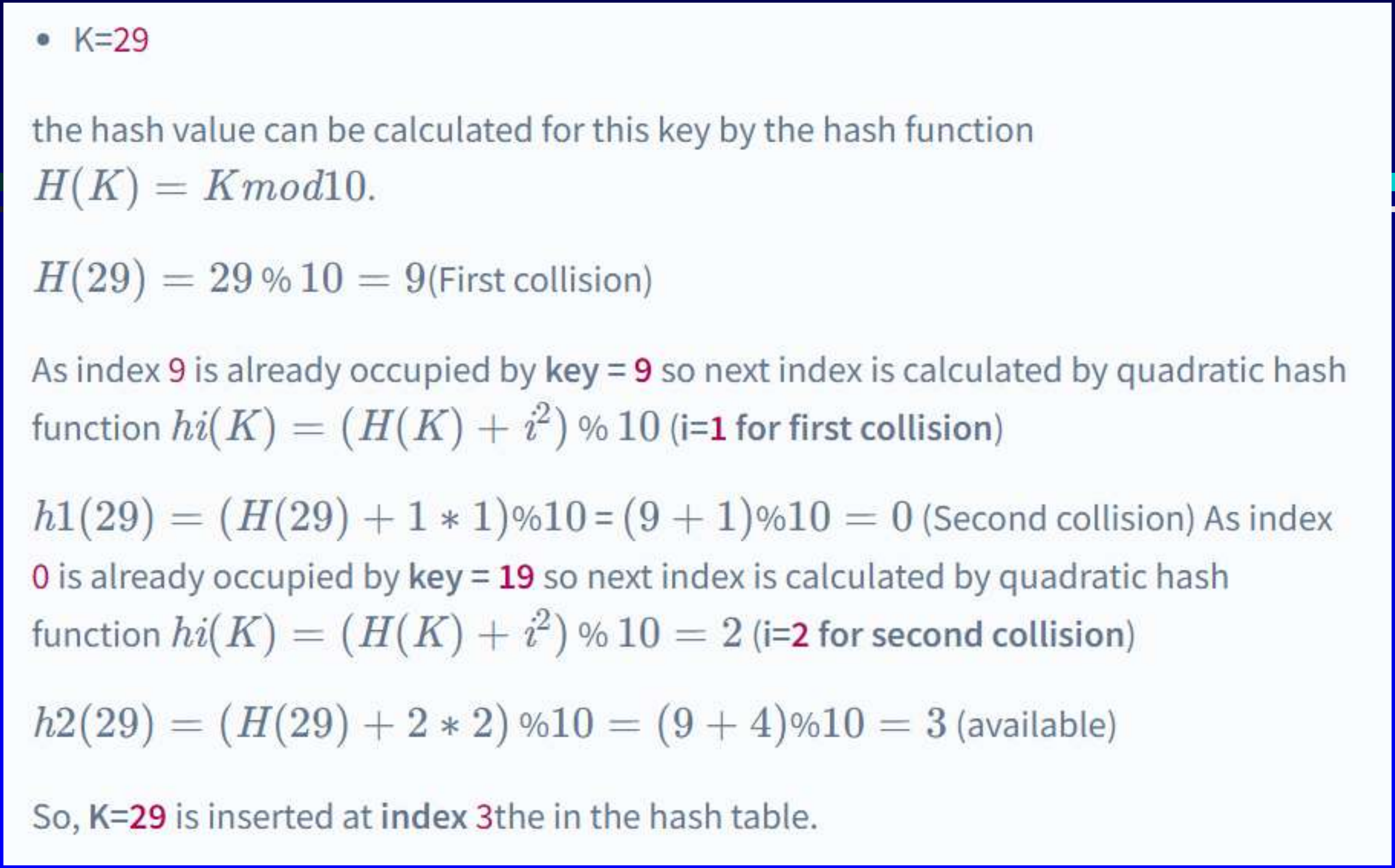
- K=19

the hash value can be calculated for this key by the hash function $H(K) = K mod 10$.

$H(19) = 19 \% 10 = 9$ (First collision) As index 9 is already occupied by **key = 9** so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10$ (**i=1 for first collision**)

$h1(19) = (H(19) + 1 * 1)\%10 = (9 + 1)\%10 = 0$ (this index position is available in the hash table) So, K=19 is inserted at index 0 in the hash table as shown below

index	keys
0	19
1	
2	
3	
4	
5	
6	
7	
8	
9	9



- K=29

the hash value can be calculated for this key by the hash function $H(K) = Kmod10.$

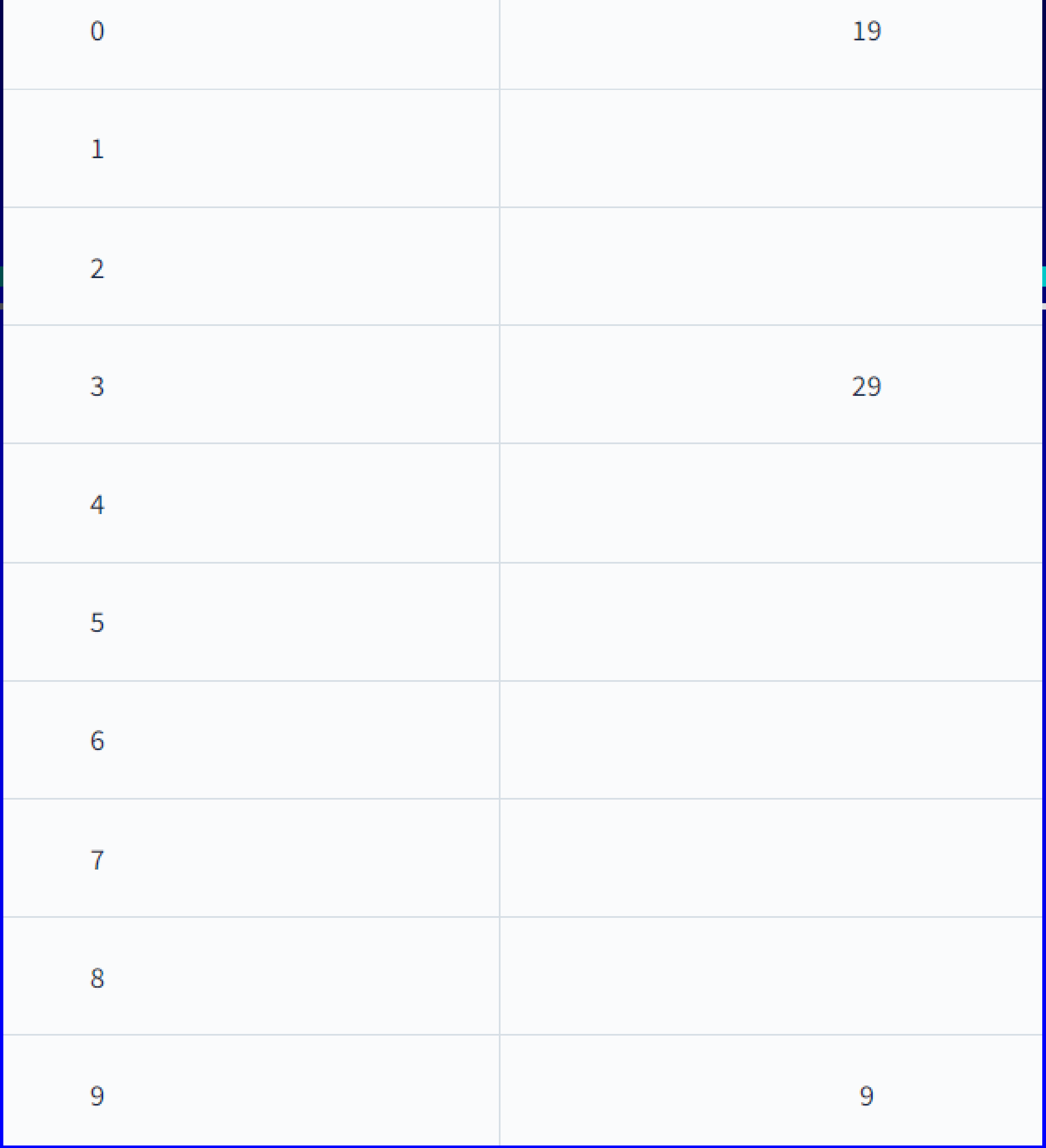
$H(29) = 29 \% 10 = 9$(First collision)

As index 9 is already occupied by **key = 9** so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10$ (**i=1 for first collision**)

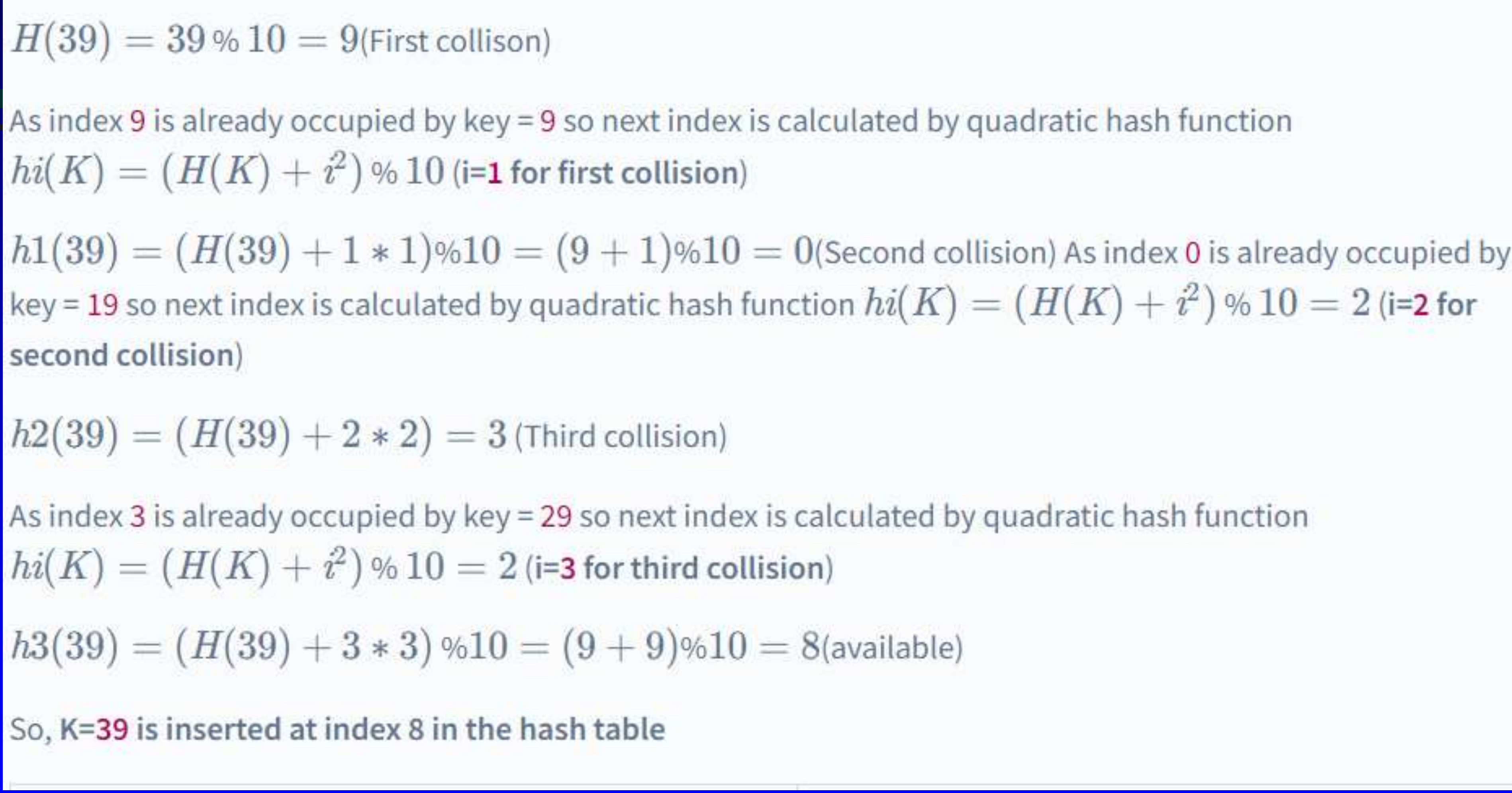
$h1(29) = (H(29) + 1 * 1)$%10 = $(9 + 1)$%10 $= 0$ (Second collision) As index 0 is already occupied by **key = 19** so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10 = 2$ (**i=2 for second collision**)

$h2(29) = (H(29) + 2 * 2) \%10 = (9 + 4)\%10 = 3$ (available)

So, **K=29** is inserted at **index 3**the in the hash table.



0	19
1	
2	
3	29
4	
5	
6	
7	
8	
9	9



$H(39) = 39 \% 10 = 9$(First collison)

As index 9 is already occupied by key = 9 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10$ (**i=1 for first collision**)

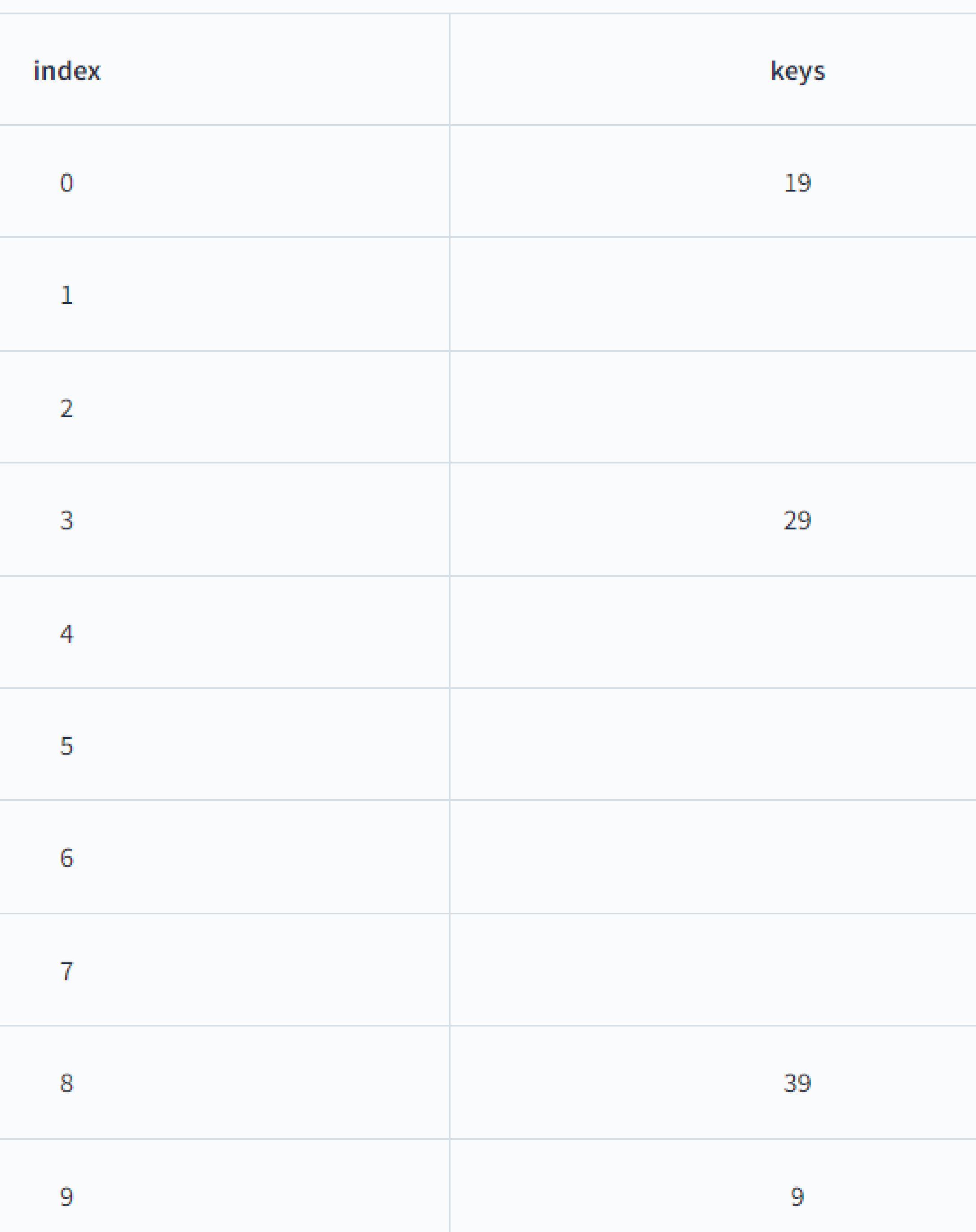
$h1(39) = (H(39) + 1 * 1)\%10 = (9 + 1)\%10 = 0$(Second collision) As index 0 is already occupied by key = 19 so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10 = 2$ (**i=2 for second collision**)

$h2(39) = (H(39) + 2 * 2) = 3$ (Third collision)

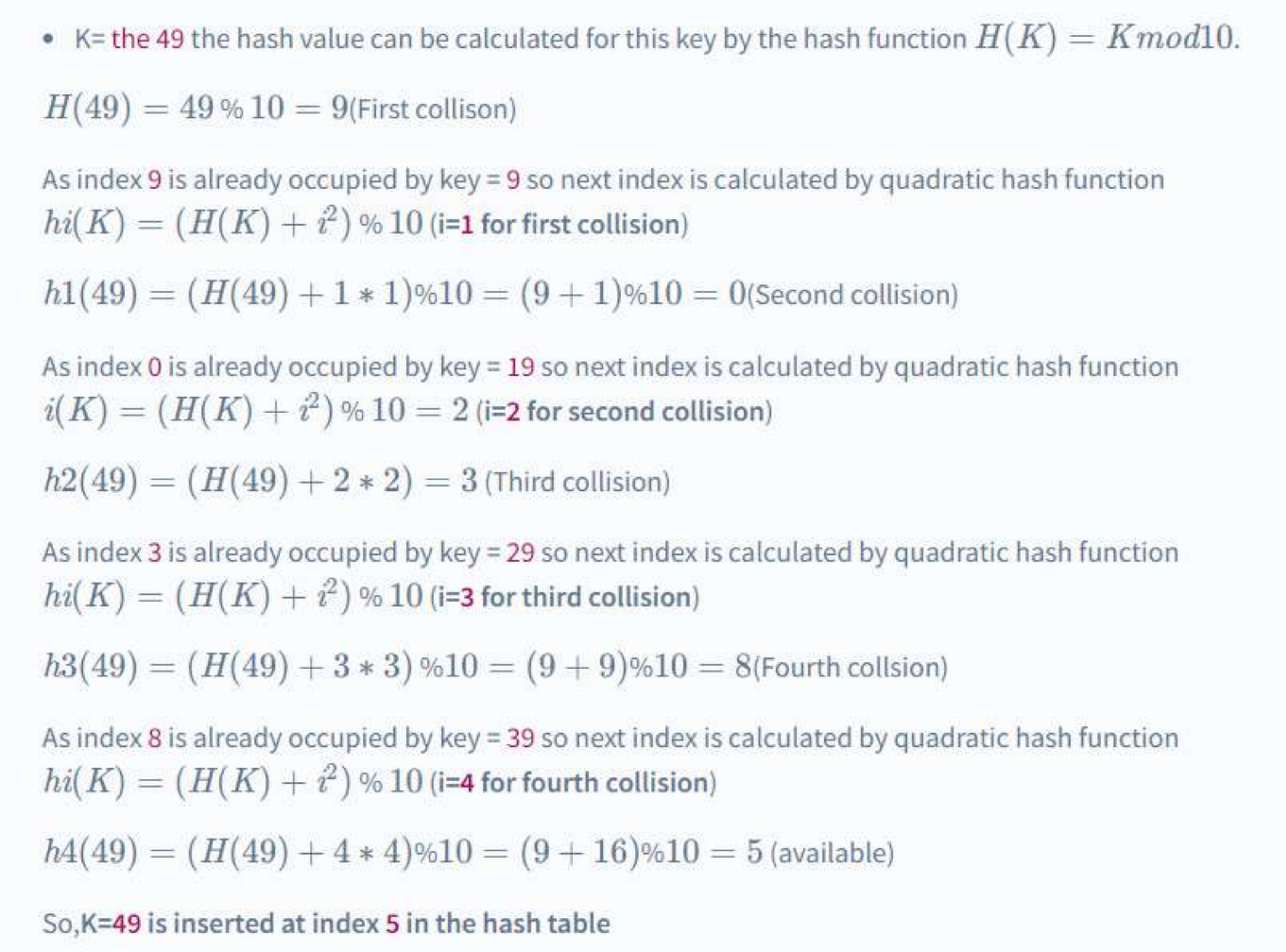
As index 3 is already occupied by key = 29 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10 = 2$ (**i=3 for third collision**)

$h3(39) = (H(39) + 3 * 3) \%10 = (9 + 9)\%10 = 8$(available)

So, **K=39 is inserted at index 8 in the hash table**



index	keys
0	19
1	
2	
3	29
4	
5	
6	
7	
8	39
9	9



- K= the 49 the hash value can be calculated for this key by the hash function $H(K) = K mod 10$.

$H(49) = 49 \% 10 = 9$(First collison)

As index 9 is already occupied by key = 9 so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10$ (**i=1 for first collision**)

$h1(49) = (H(49) + 1 * 1)\%10 = (9 + 1)\%10 = 0$(Second collision)

As index 0 is already occupied by key = 19 so next index is calculated by quadratic hash function $i(K) = (H(K) + i^2) \% 10 = 2$ (**i=2 for second collision**)

$h2(49) = (H(49) + 2 * 2) = 3$ (Third collision)

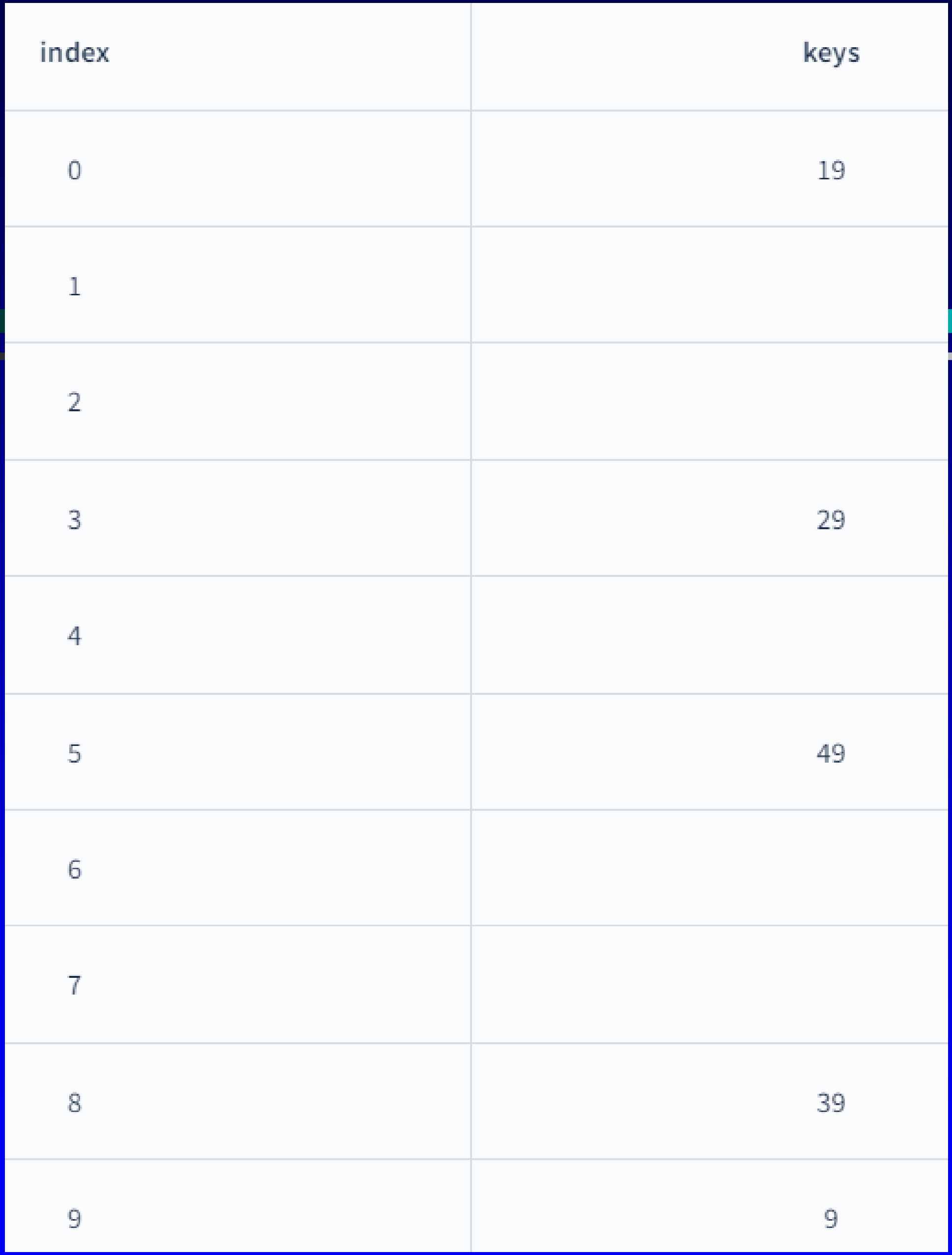
As index 3 is already occupied by key = 29 so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10$ (**i=3 for third collision**)

$h3(49) = (H(49) + 3 * 3) \%10 = (9 + 9)\%10 = 8$(Fourth collsion)

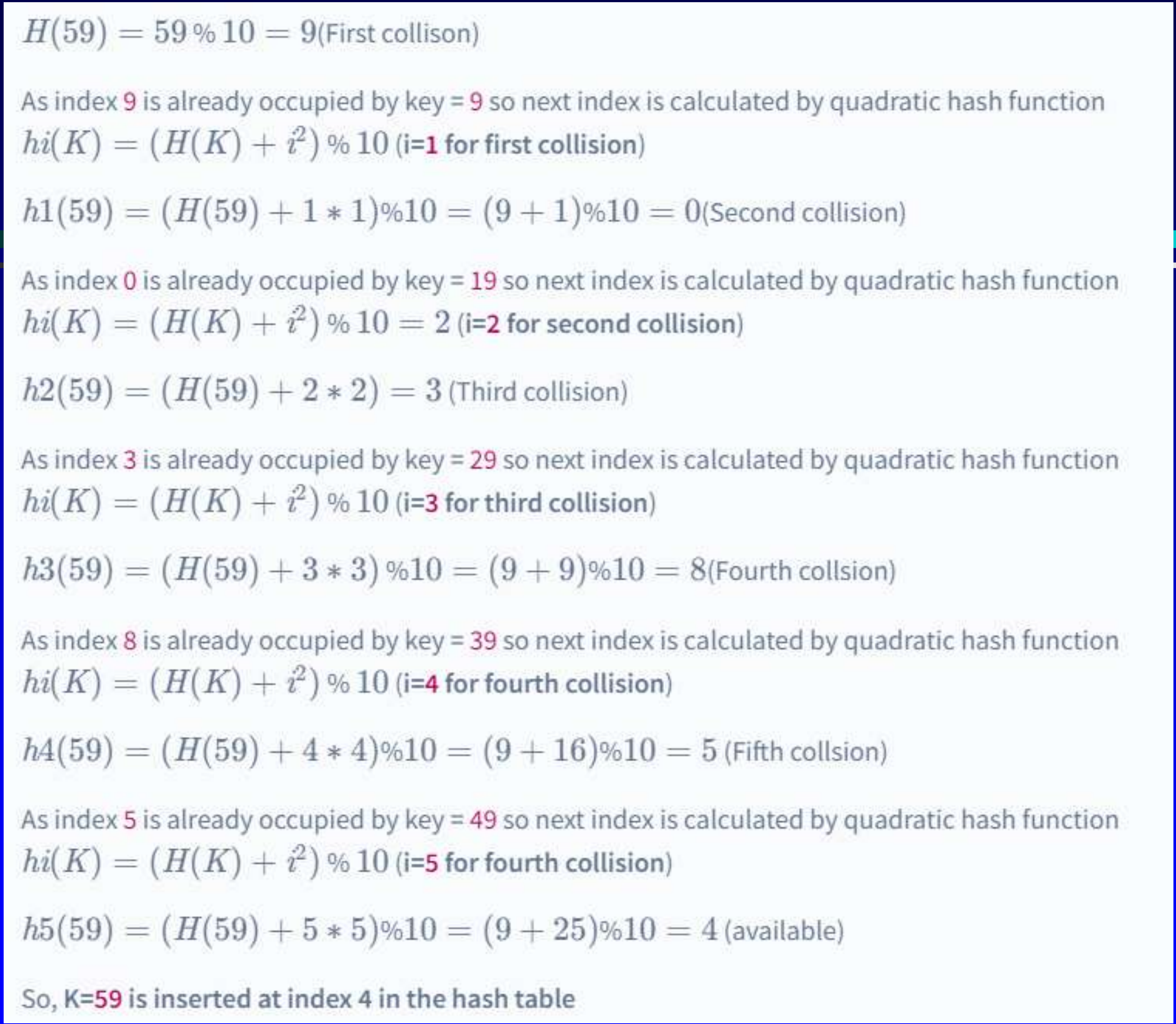
As index 8 is already occupied by key = 39 so next index is calculated by quadratic hash function $hi(K) = (H(K) + i^2) \% 10$ (**i=4 for fourth collision**)

$h4(49) = (H(49) + 4 * 4)\%10 = (9 + 16)\%10 = 5$ (available)

So,**K=49 is inserted at index 5 in the hash table**



index	keys
0	19
1	
2	
3	29
4	
5	49
6	
7	
8	39
9	9



$H(59) = 59 \% 10 = 9$(First collison)

As index 9 is already occupied by key = 9 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10$ (**i=1 for first collision**)

$h1(59) = (H(59) + 1 * 1)\%10 = (9 + 1)\%10 = 0$(Second collision)

As index 0 is already occupied by key = 19 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10 = 2$ (**i=2 for second collision**)

$h2(59) = (H(59) + 2 * 2) = 3$ (Third collision)

As index 3 is already occupied by key = 29 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10$ (**i=3 for third collision**)

$h3(59) = (H(59) + 3 * 3) \%10 = (9 + 9)\%10 = 8$(Fourth collsion)

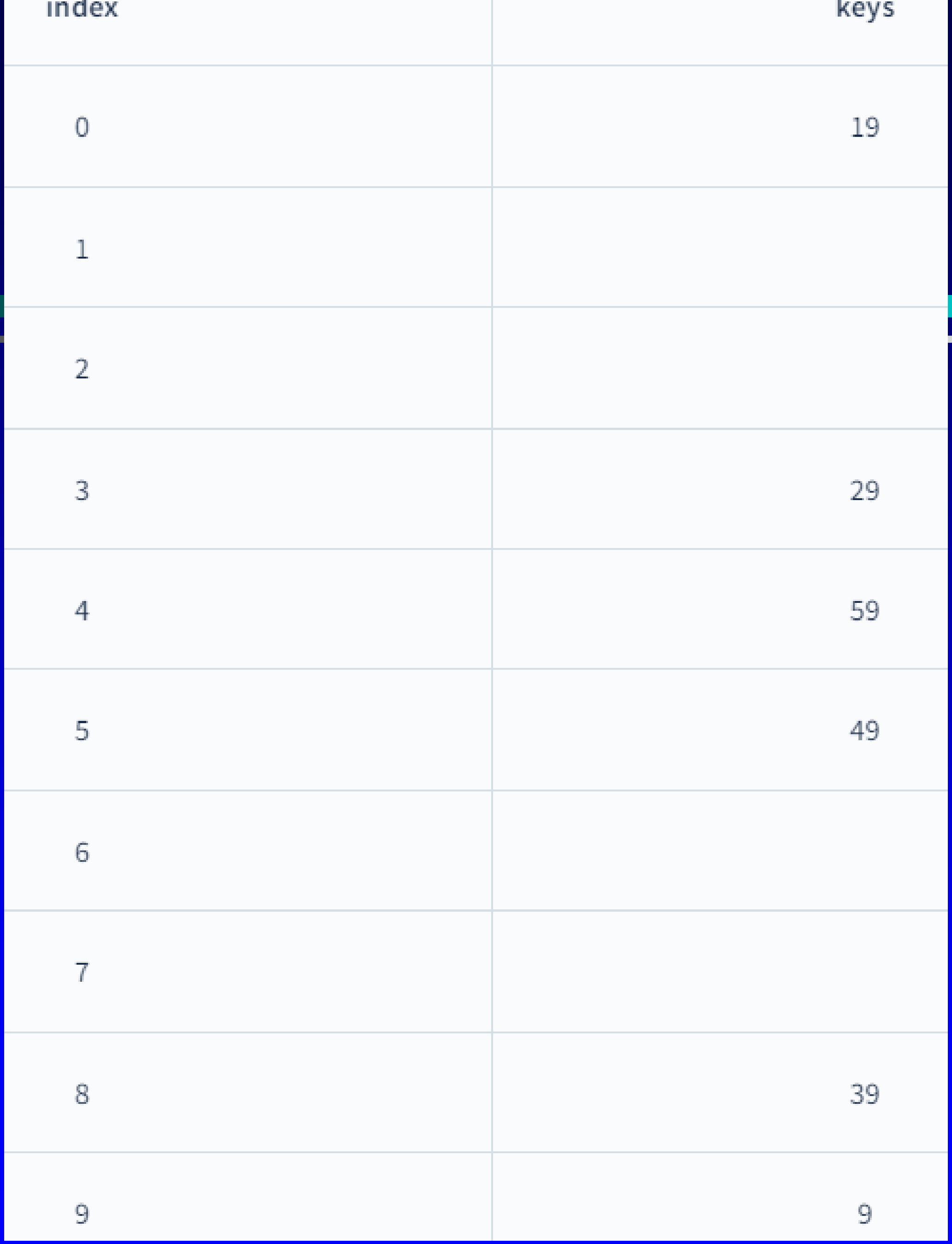
As index 8 is already occupied by key = 39 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10$ (**i=4 for fourth collision**)

$h4(59) = (H(59) + 4 * 4)\%10 = (9 + 16)\%10 = 5$ (Fifth collsion)

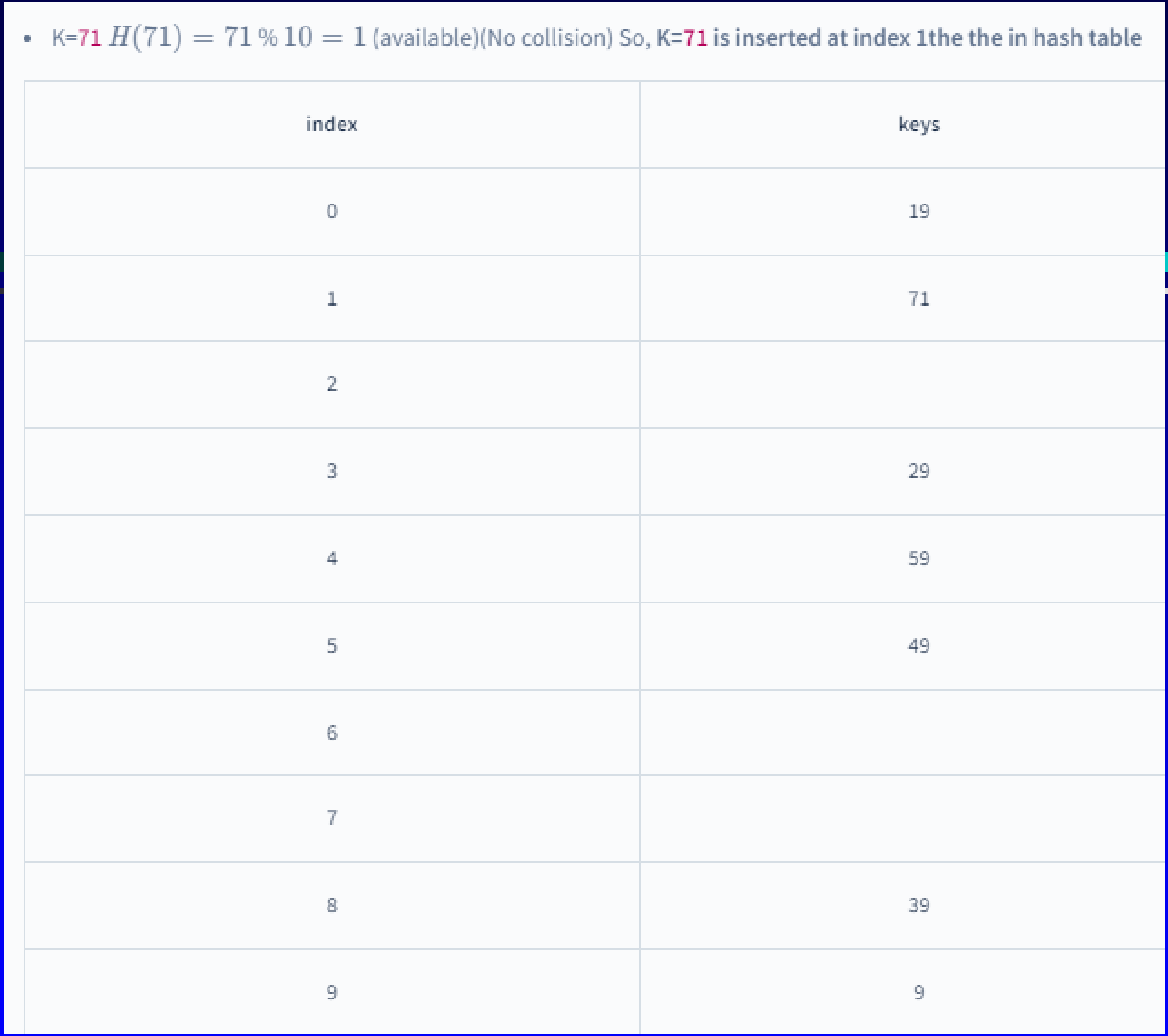
As index 5 is already occupied by key = 49 so next index is calculated by quadratic hash function
$hi(K) = (H(K) + i^2) \% 10$ (**i=5 for fourth collision**)

$h5(59) = (H(59) + 5 * 5)\%10 = (9 + 25)\%10 = 4$ (available)

So, **K=59 is inserted at index 4 in the hash table**



index	keys
0	19
1	
2	
3	29
4	59
5	49
6	
7	
8	39
9	9



- K=71 $H(71) = 71 \% 10 = 1$ (available)(No collision) So, K=71 **is inserted at index 1the the in hash table**

index	keys
0	19
1	71
2	
3	29
4	59
5	49
6	
7	
8	39
9	9