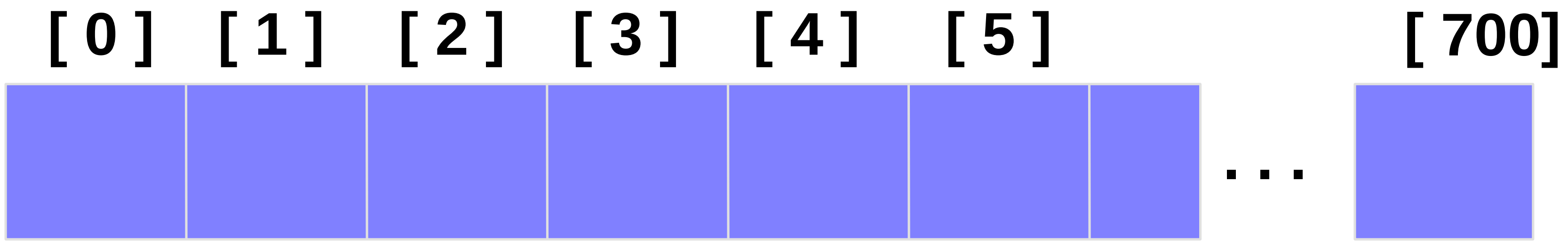
Unit-5
Sorting and Searching

Course: MCA

Subject: Data and File Structure

What is a Hash Table ?

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

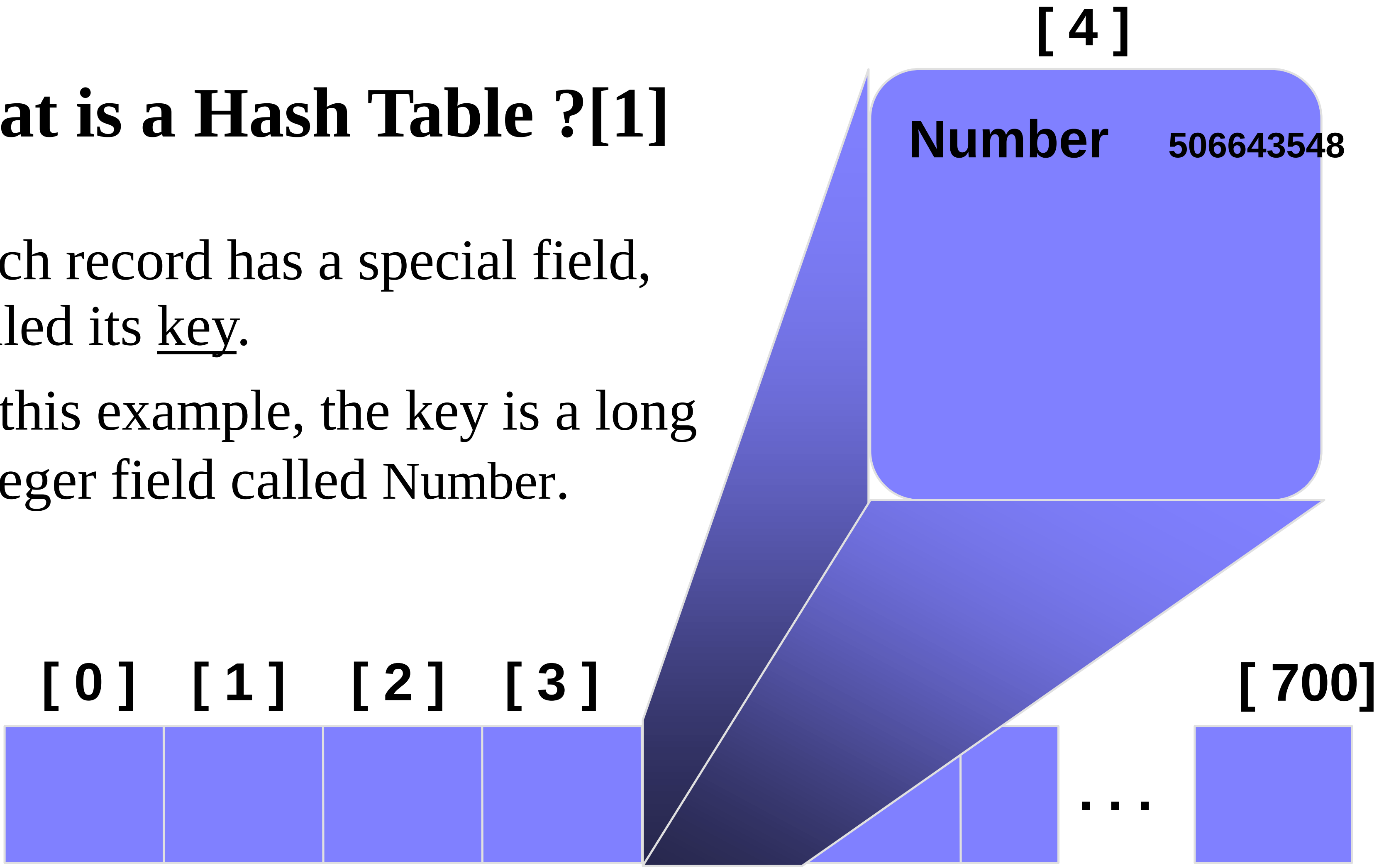


[0] [1] [2] [3] [4] [5] . . . [700]

An array of records

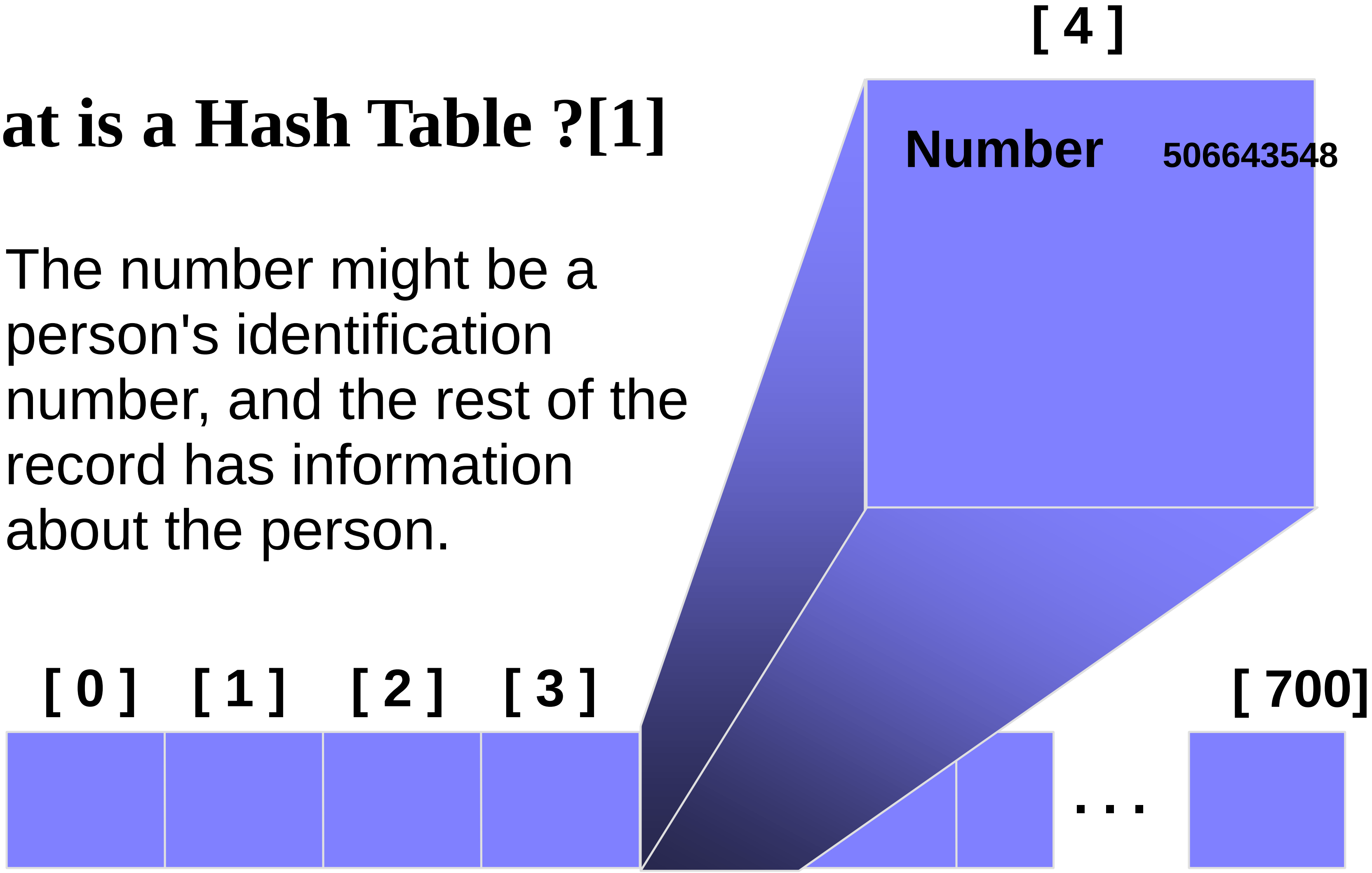
What is a Hash Table ?[1]

- 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 ?[1]

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



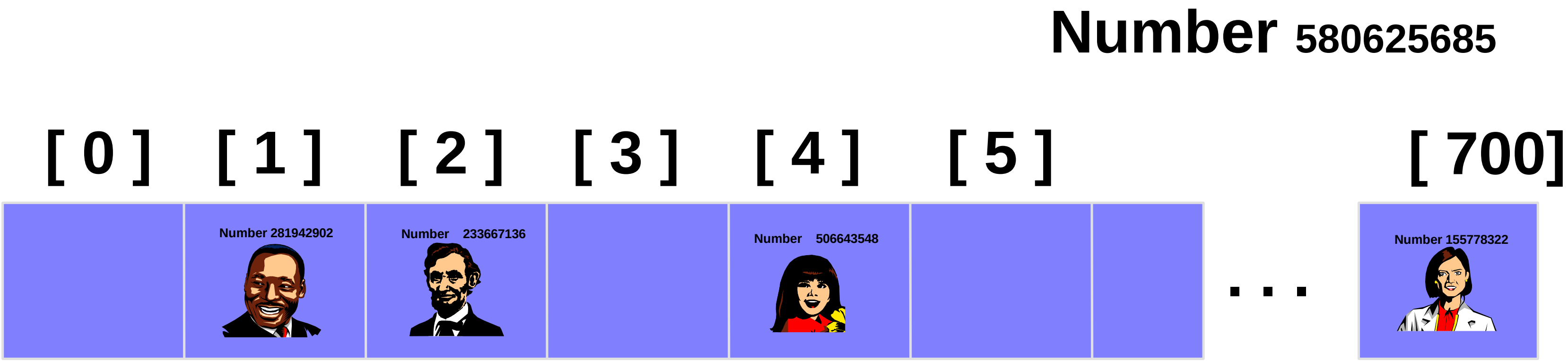
[4]

Number 506643548

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

Inserting a New Record[2]

- 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.



Number 580625685

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

Number 281942902

Number 233667136

Number 506643548

. . .

Number 155778322

Inserting a New Record[2]

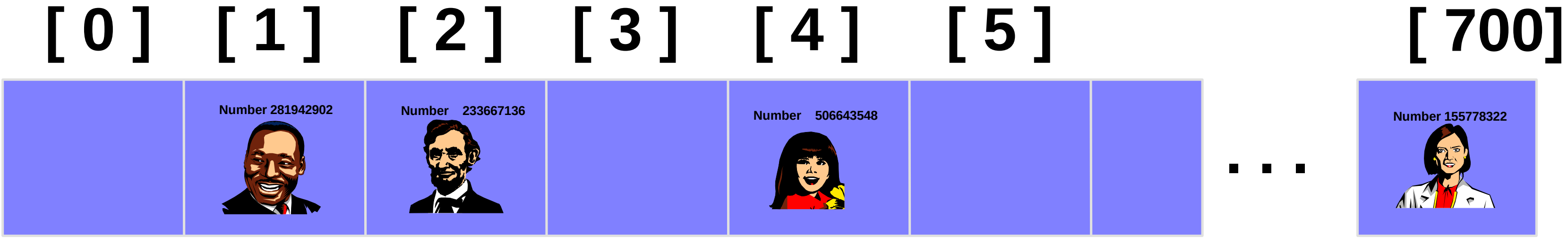
Number 580625685

- Typical way to create a hash value:

(Number mod 701)

What is (580625685 mod 701) ?

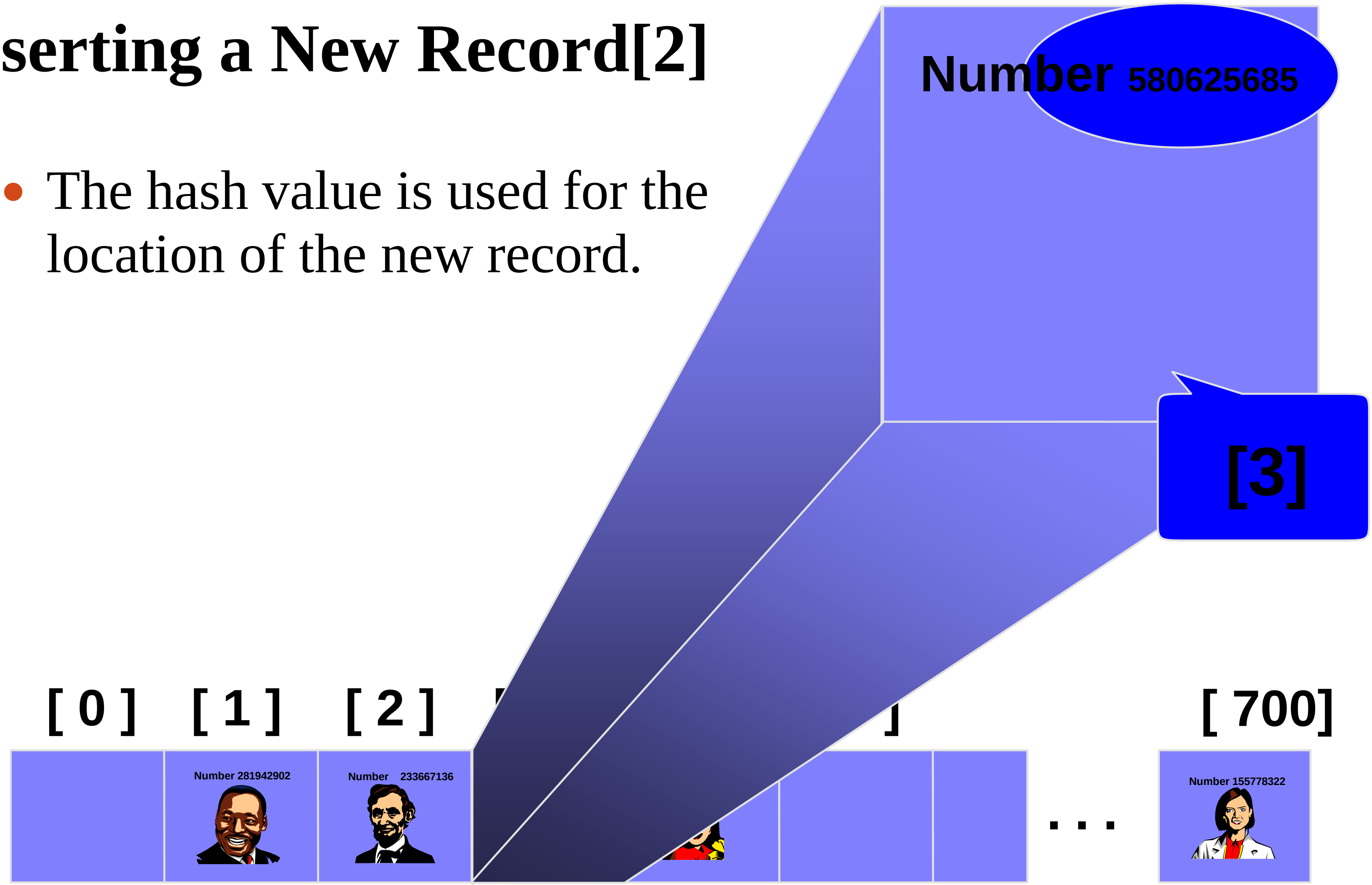
3



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

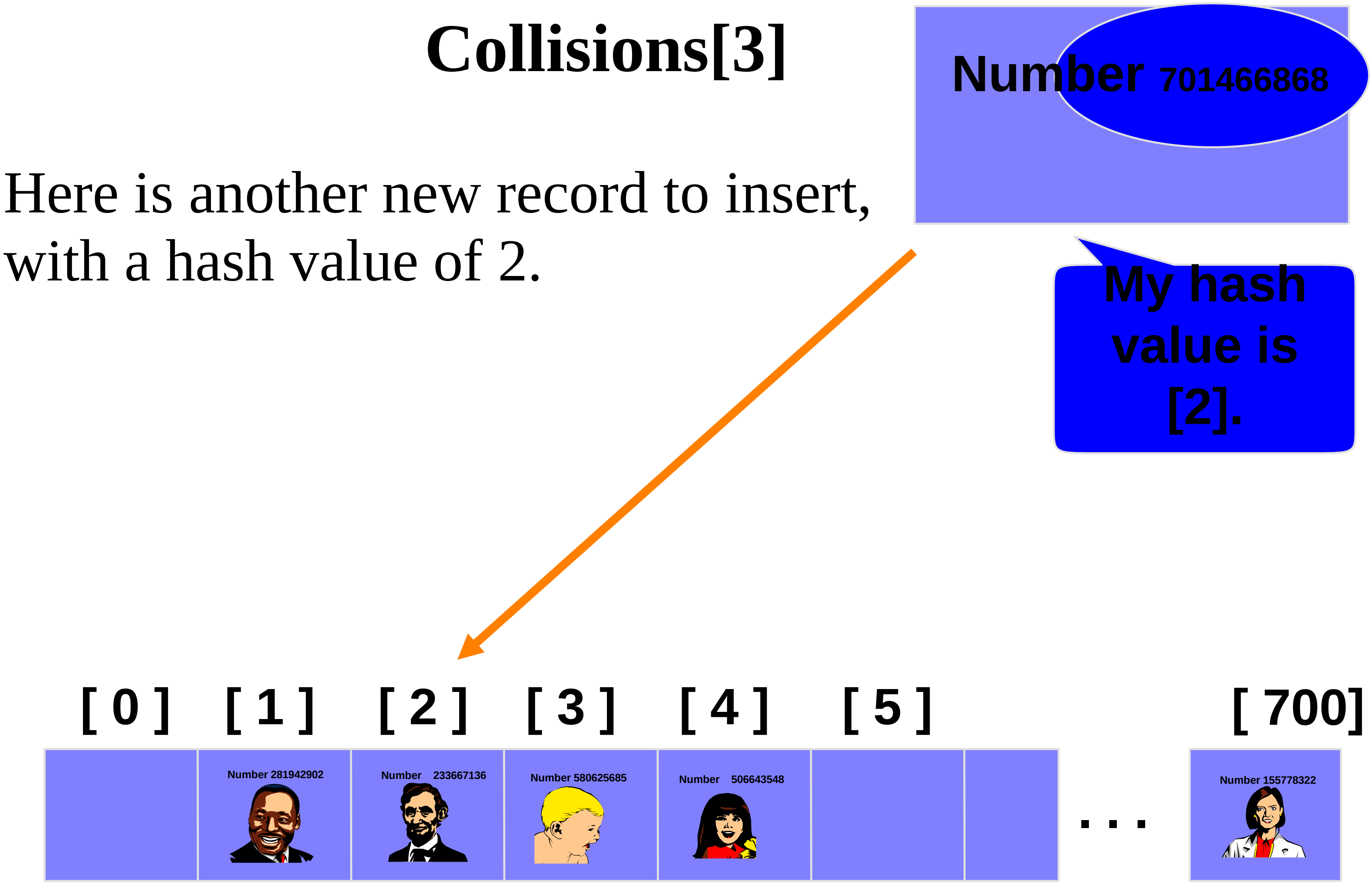
Inserting a New Record[2]

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



Collisions[3]

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



Number 701466868

My hash value is [2].

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

Number 281942902

Number 233667136

Number 580625685

Number 506643548

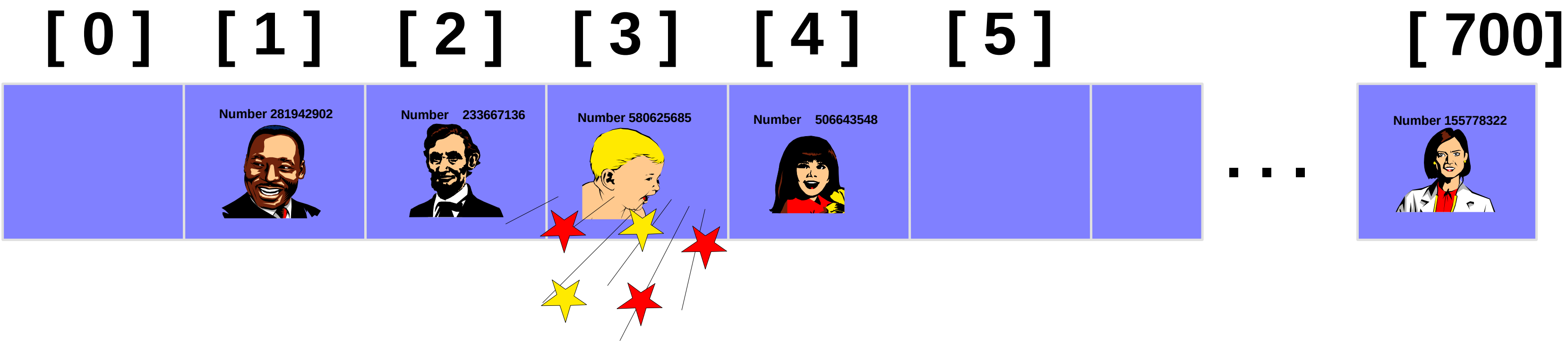
. . .

Number 155778322

Collisions

Number 701466868

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



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

Number 281942902

Number 233667136

Number 580625685

Number 506643548

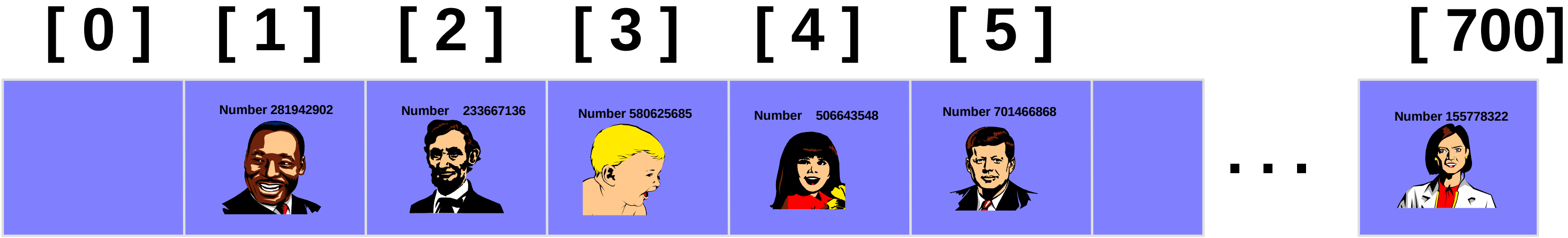
. . .

Number 155778322

Collisions

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

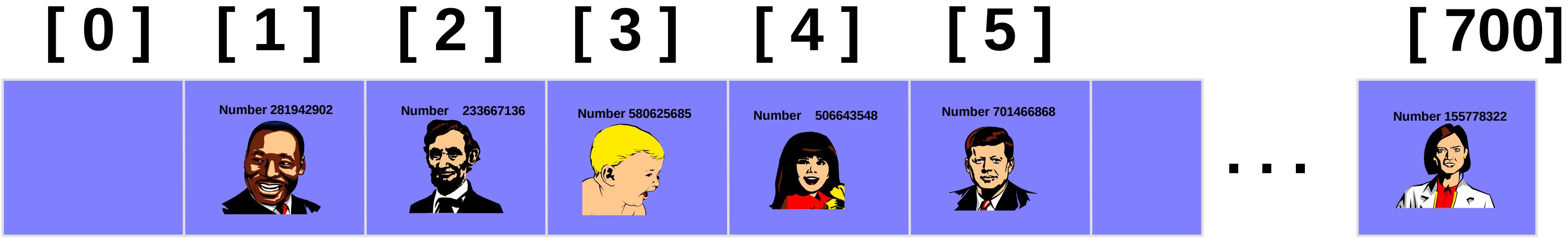
The new record goes
in the empty spot.



Searching for a Key

Number 701466868

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

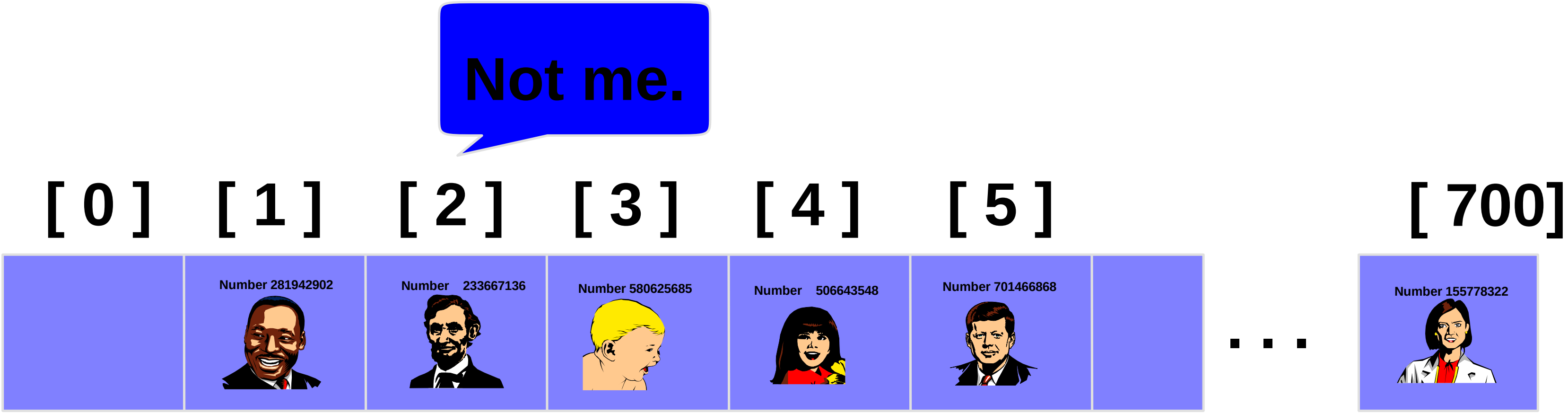


Searching for a Key

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

Number 701466868

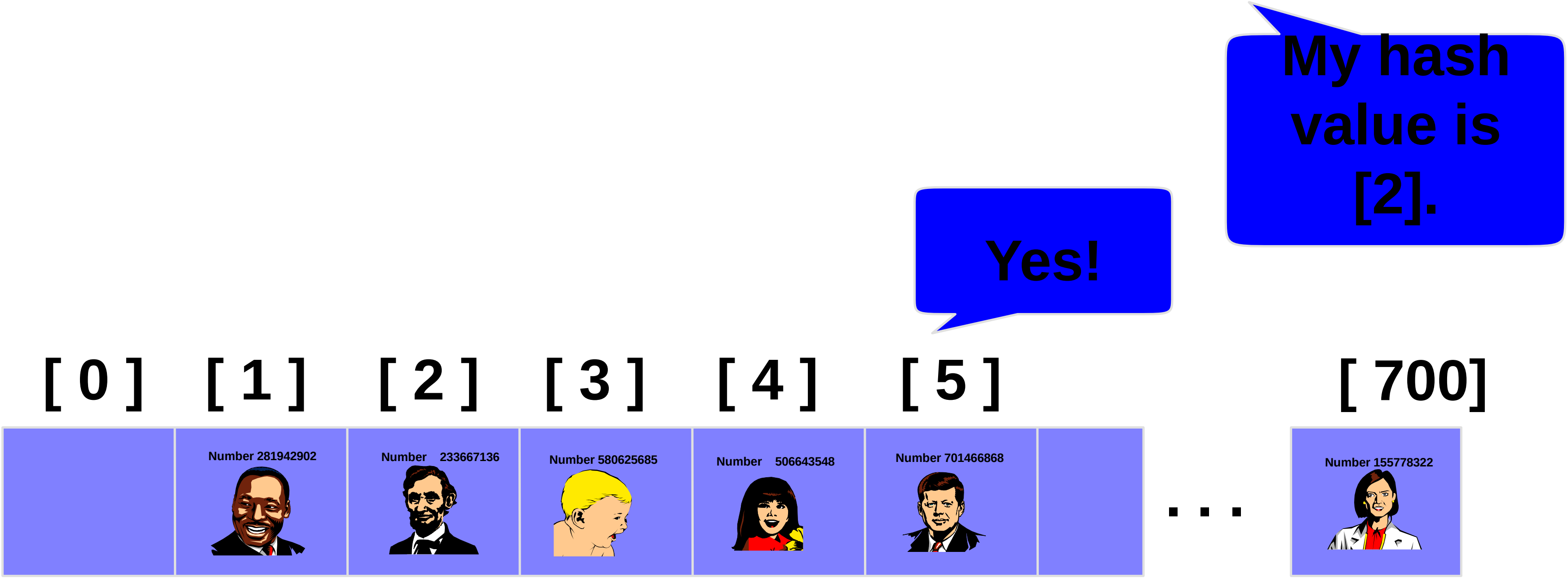
My hash value is [2].



Searching for a Key

Number 701466868

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



My hash value is [2].

Yes!

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

Number 281942902

Number 233667136

Number 580625685

Number 506643548

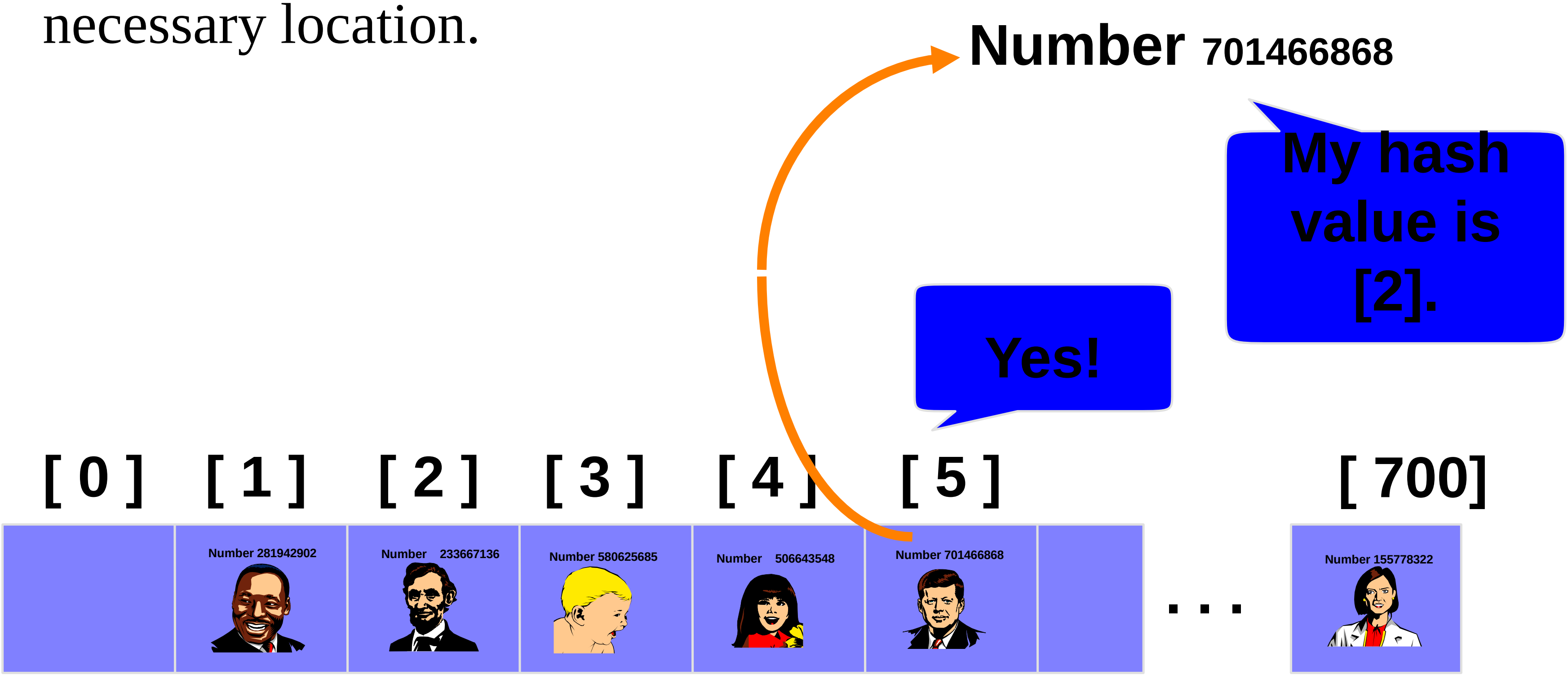
Number 701466868

. . .

Number 155778322

Searching for a Key

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



Number 701466868

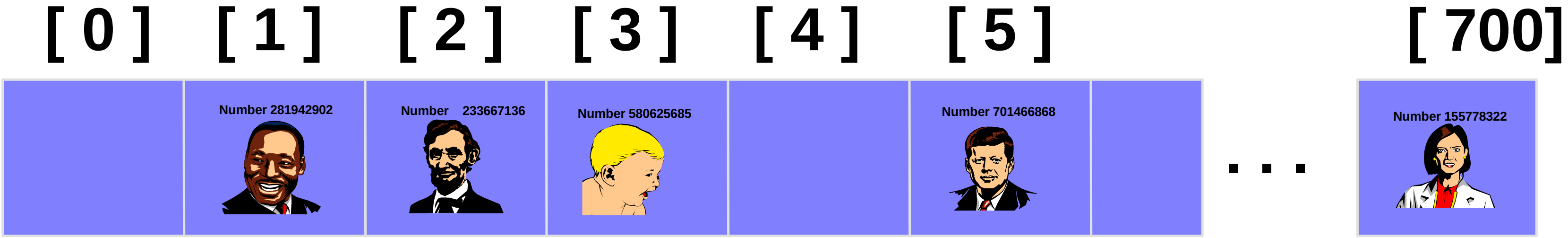
My hash value is [2].

Yes!

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

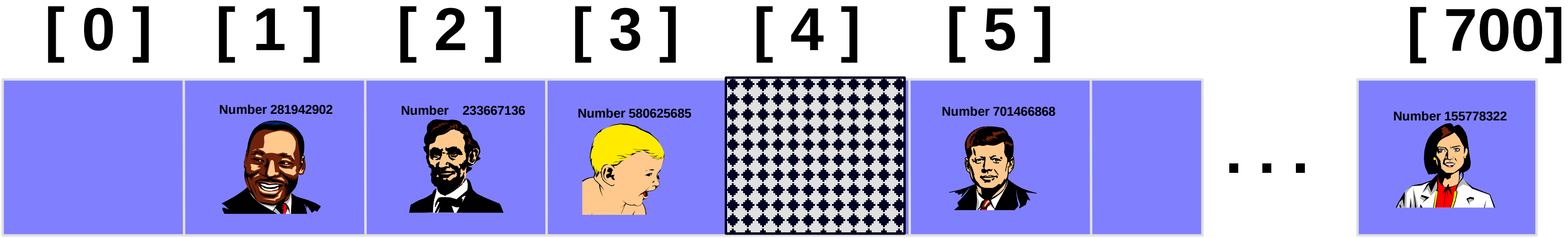
Deleting a Record

- 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.



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.



Handling collisions

- If the number of possible keys greatly exceeds the numbers of records, and of computed storage locations, hash collisions become inevitable and so have to be handled without loss of data.

- 3 approaches are used to handle collisions:

- open hashing
- quadratic hashing
- chained hashing

Files

A file can be seen as

1. A stream of bytes (no structure), or
2. A collection of records with fields

A Stream File[4]

- File is viewed as a sequence of bytes:

```
87359CarrollAlice in wonderland38180FolkFile Structures ...
```

- Data semantics is lost: there is no way to get it apart again.

Field and Record Organization

Definitions

Record: a collection of related fields.

Field: the smallest logically meaningful unit of information in a file.

Key: a subset of the fields in a record used to identify (uniquely) the record.

e.g. In the example file of books:

- Each line corresponds to a record.
- Fields in each record: ISBN, Author, Title

Record Keys

- **Primary key**: a key that uniquely identifies a record.
- **Secondary key**: other keys that may be used for search
  - Author name
  - Book title
  - Author name + book title
- Note that in general not every field is a key (keys correspond to fields, or a combination of fields, that may be used in a search).

Field Structures

- Fixed-length fields

```
87359Carroll   Alice in wonderland
38180Folk      File Structures
```

- Begin each field with a length indicator

```
058735907Carroll19Alice in wonderland
053818004Folk15File Structures
```

- Place a delimiter at the end of each field

```
87359|Carroll|Alice in wonderland|
38180|Folk|File Structures|
```

- Store field as keyword = value

```
ISBN=87359|AU=Carroll|TI=Alice in wonderland|
ISBN=38180|AU=Folk|TI=File Structures
```

Record Structures

1. Fixed-length records.
2. Fixed number of fields.
3. Begin each record with a length indicator.
4. Use an index to keep track of addresses.
5. Place a delimiter at the end of the record.

Fixed-length records[4]

Two ways of making fixed-length records:

1. Fixed-length records with fixed-length fields.

87359	Carroll	Alice in wonderland
03818	Folk	File Structures

2. Fixed-length records with variable-length fields.

87359|Carroll|Alice in wonderland|	*unused*
38180|Folk|File Structures|	*unused*

Variable-length records[5]

- Fixed number of fields:

```
87359|Carroll|Alice in wonderland|38180|Folk|File Structures| ...
```

- Record beginning with length indicator:

```
3387359|Carroll|Alice in wonderland|2638180|Folk|File Structures| ..
```

- Use an index file to keep track of record addresses:
  - The index file keeps the byte offset for each record; this allows us to search the index (which have fixed length records) in order to discover the beginning of the record.
- Placing a delimiter: e.g. end-of-line char

File Operations

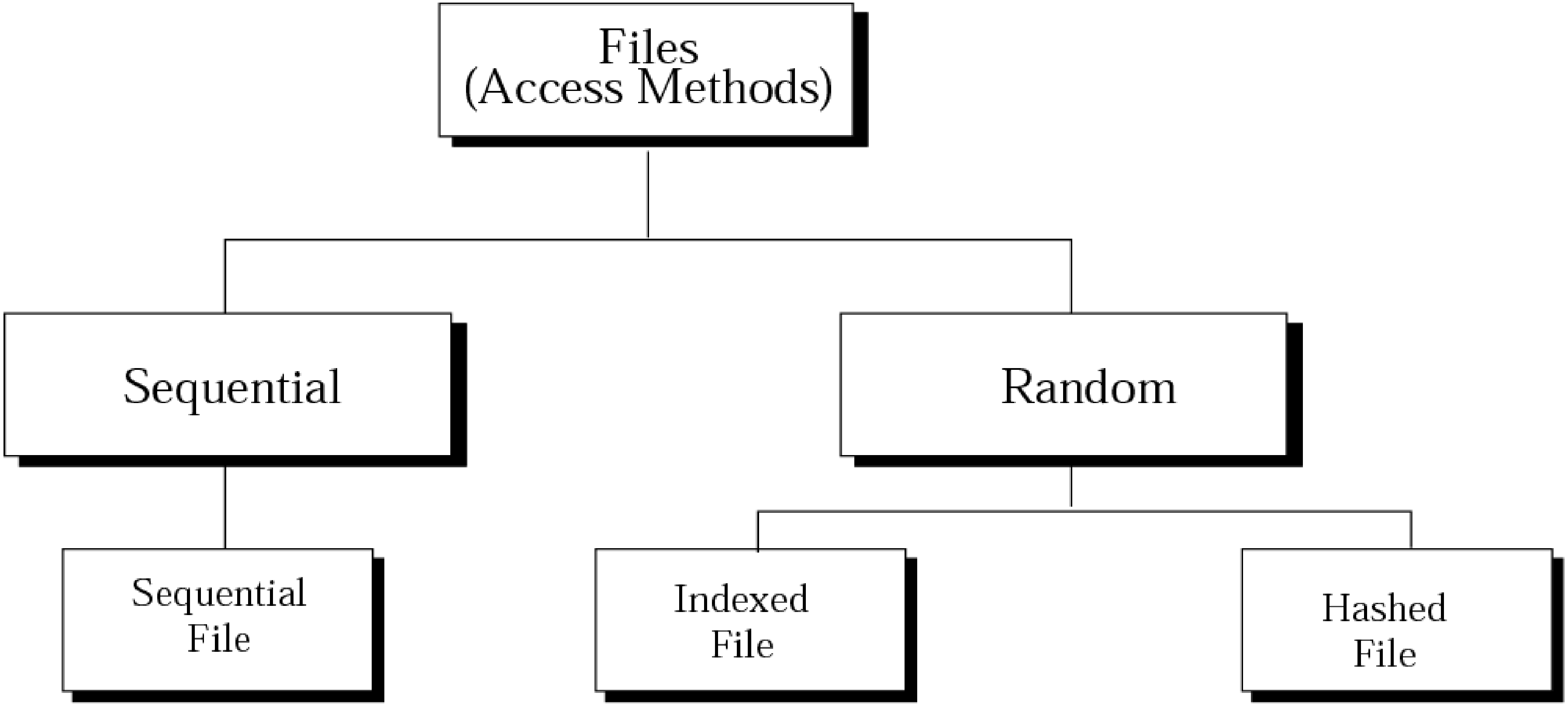
- Typical Operations:
  - Retrieve a record
  - Insert a record
  - Delete a record
  - Modify a field of a record

- In direct files:
  - Get a record with a given field value

- In sequential files:
  - Get the next record

Taxonomy of file structures[6]

- The access method determines how records can be retrieved: sequentially or randomly.



- One record after another, from beginning to end
- Access one specific record without having to retrieve all records before it

Sequential file

- Sequential file –Records can only be accessed sequentially, one after another, from beginning to end.

- Records are arranged in sequential manner.

Mapping in an indexed file

- To access a record in a file Randomly, you need to know the address of the record.

- An index file can relate the key to the record address.

Indexed files

- An index file is made of a data file, which is a sequential file, and an index.
- Index – a small file with only two fields:
  - The key of the sequential file
  - The address of the corresponding record on the disk.
- To access a record in the file :
  - Load the entire index file into main memory.
  - Search the index file to find the desired key.
  - Retrieve the address the record.
  - Retrieve the data record. (using the address)
- Inverted file –
  you can have more than one index, each with a different key.

Mapping in a hashed file

- A hashed file uses a hash function to map the key to the address.
- Eliminates the need for an extra file (index).
- There is no need for an index and all of the overhead associated with it.

Direct Hashing

- The file must contain a record for every possible key.
- Adv. – no collision.
- Disadv. – space is wasted.

- Hashing techniques –
  map a large population of possible keys into
  a small address space.

References

1. An introduction to Datastructure with application by jean Trembley and sorrenson
2. Data structures by schaums and series –seymour lipschutz
3. http://en.wikipedia.org/wiki/Book:Data_structures
4. http://www.amazon.com/Data-Structures-Algorithms
5. http://www.amazon.in/Data-Structures-Algorithms-Made-Easy/dp/0615459811/
6. http://www.amazon.in/Data-Structures-SIE-Seymour-Lipschutz/dp

List of images

1. http://www.expertsmind.com/questions/depth-first-search-30153061.aspx

2. http://burtleburtle.net/bob/hash/index.html

3.http://herakles.zcu.cz/~skala/PUBL/PUBL_2010/2010_WSEASCorfu_Hash-final.pdf

4. http://www.scribd.com/doc/199819816/Fundamental-Data-Structures

5. http://searchsqlserver.techtarget.com/definition/hashing

6. http://searchsqlserver.techtarget.com/definition/hashing