1.3.1 Uses of Hashing

Hash Tables

A hash table is a data structure that stores data using a key to determine where the data is stored.

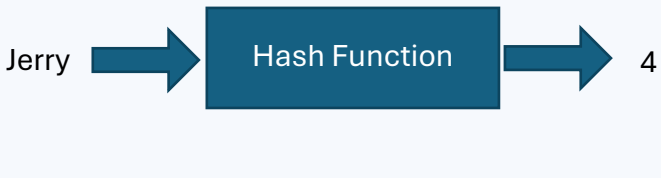
A hashing algorithm is applied to the key and returns a hash value. This hash value is then used as an index to locate where the data item should be stored in the table.

Example:

This hash table contains the names and mobile numbers of a person's contacts.

The data Jerry 07847564736 is being inserted into the hash table.

The key is the person's name "Jerry" and is input into the hash algorithm to produce a hash.



Index	Data
1	
2	Nigel 07943443221
3	
4	Jerry 07847564736
5	
…	

The result of applying the hash algorithm to the key "Jerry" was location 4 so Jerry's information was inserted in location 4 of the hash table.

Pros:

- Very efficient at inserting new entries to the table.
- Very efficient at searching for entries in the table.
- O(1) average time complexity

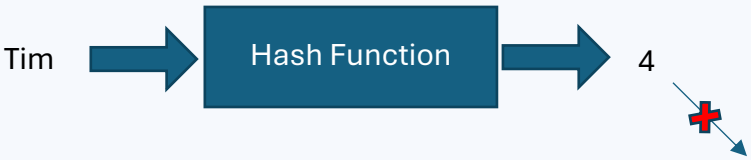
Collisions

A collision occurs when **two or more keys produce the same hash value**.

Hash tables can become less efficient when collisions occur frequently. A poor hash function, or a table that is too full, can cause many keys to map to the same locations, increasing the time needed to find or insert data.

Example:

The key "Tim" has produced a hash for location 4 but Jerry's data is already contained in this location.



Index	Data
1	
2	Nigel 07943443221
3	
4	Jerry 07847564736
5	
…	

Collision Handling

In the event of a collision, collision handling methods can be applied to decide what should be done with the data that is being entered into the table.

Chaining

If there is a collision, a list is created in that slot and any additional items are appended to the list.

Linear Probing

If there is a collision, check the next location until a free location is found.

Overflow Area

If there is a collision, put the data in a reserved area of memory called the overflow area.

Password Storage Using Hashing

Hashing can be used to securely store passwords on a device or server.

Saving a password on a device or server is unsafe as the file holding the passwords could be accessed.

By hashing the passwords on entry and storing the hash value in the user database the original password does not need to be stored.

Unlike encryption, hashing cannot be reversed and so even if the password file on the server is hacked, the hacker would only get the hashed password and not the actual password.

Process

1. Computer A is trying to log in to their account on a server.
2. Computer A Enters username and password and runs the password through a hash function.
3. The username and hashed password are transmitted to the server.
4. The server checks the hashed password against the hash value stored in the database.