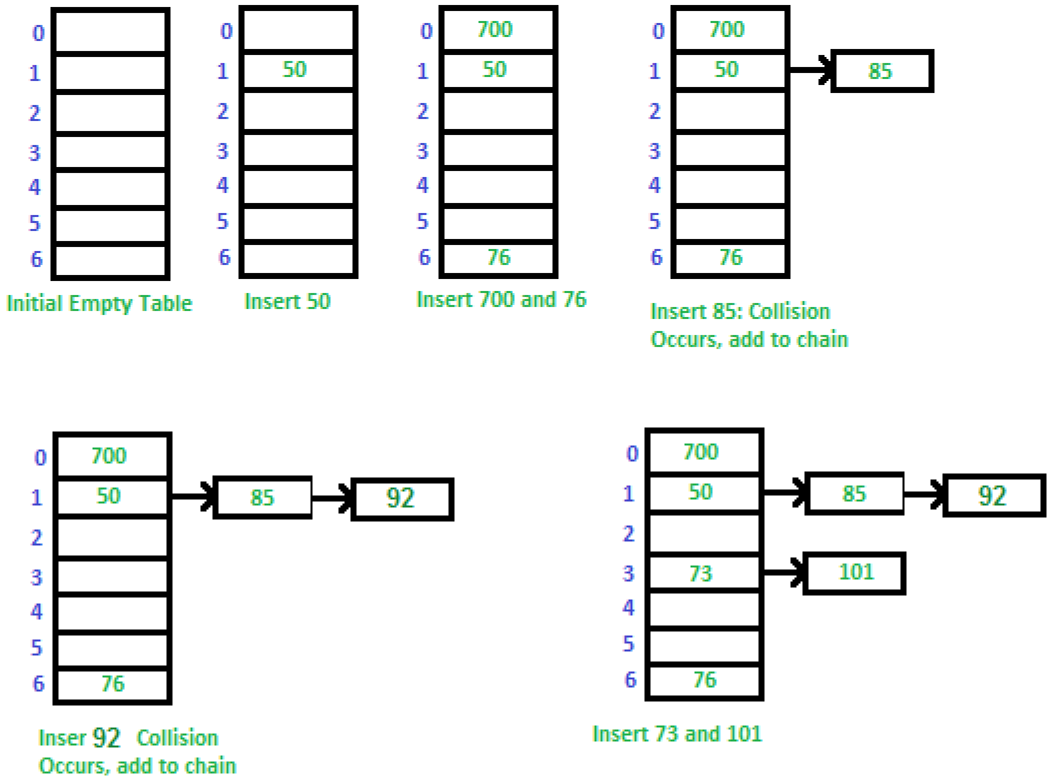
Separate Chaining:

The idea behind separate chaining is to implement the array as a linked list called a chain. Separate chaining is one of the most popular and commonly used techniques in order to handle collisions.

*The **linked list** data structure is used to implement this technique. So what happens is, when multiple elements are hashed into the same slot index, then these elements are inserted into a singly-linked list which is known as a chain.*

Here, all those elements that hash into the same slot index are inserted into a linked list. Now, we can use a key K to search in the linked list by just linearly traversing. If the intrinsic key for any entry is equal to K then it means that we have found our entry. If we have reached the end of the linked list and yet we haven't found our entry then it means that the entry does not exist. Hence, the conclusion is that in separate chaining, if two different elements have the same hash value then we store both the elements in the same linked list one after the other.

Example: Let us consider a simple hash function as "**key mod 7**" and a sequence of keys as 50, 700, 76, 85, 92, 73, 101



Initial Empty Table

Insert 50

Insert 700 and 76

Insert 85: Collision Occurs, add to chain

Inser 92 Collision Occurs, add to chain

Insert 73 and 101

Advantages:

- Simple to implement.
- Hash table never fills up, we can always add more elements to the chain.
- Less sensitive to the hash function or load factors.
- It is mostly used when it is unknown how many and how frequently keys may be inserted or deleted.

Disadvantages:

- The cache performance of chaining is not good as keys are stored using a linked list. Open addressing provides better cache performance as everything is stored in the same table.
- Wastage of Space (Some Parts of the hash table are never used)
- If the chain becomes long, then search time can become O(n) in the worst case
- Uses extra space for links