**Title:**To implement operation on hash table and create telephone directory and a Dictionary

**Objectives:**To learn implementation of hash table.

**Problem Statement:**

Consider telephone book database of N clients. Make use of a hash table implementation to quickly look up client's telephone number.

Implement all the functions of a dictionary (ADT) using hashing.
Data: Set of (key, value) pairs, Keys are mapped to values, Keys must be comparable, Keys must be unique
Standard Operations: Insert(key, value), Find(key), Delete(key)

**Outcomes:**

Understanding the implementation of hash table.

**Software & Hardware requirements:**

Open Source C++ Programming tool Eclipse.
HP 280 G1 MT 500GB HDD 4GB RAM DDR 3 INTEL CORE I5.

**Theory:**

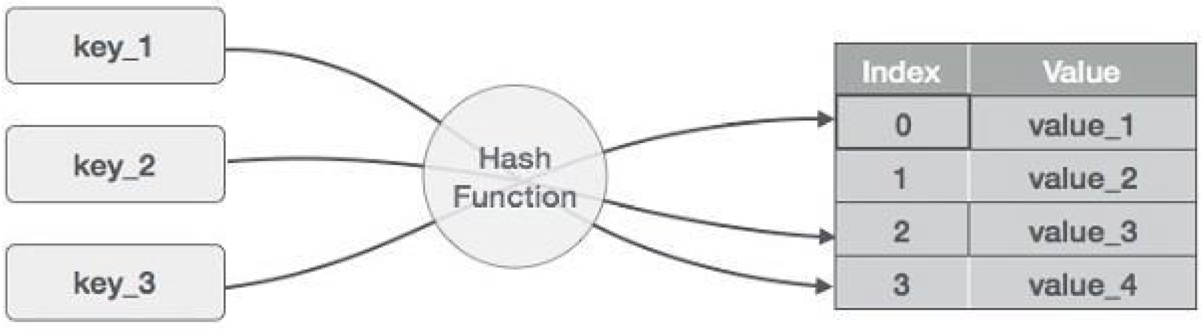
**Hash Table:**

Hash Table is a data structure which stores data in an associative manner. In a hash table, data is stored in an array format, where each data value has its own unique index value. Access of data becomes very fast if we know the index of the desired data.

Hashing:

Hashing is a technique to convert a range of key values into a range of indexes of an array. We're going to use modulo operator to get a range of key values.

Consider an example of hash table of size 20, and the following items are to be stored. Item are in the (key,value) format.



## Basic Operations

Following are the basic primary operations of a hash table.

- **Search** – Searches an element in a hash table.
- **Insert** – inserts an element in a hash table.
- **delete** – Deletes an element from a hash table.

## Algorithm:

### Algorithm : Hash Key

```
1. Start
2. Hkey = data % (size of hash table)
3. Return Hkey
4. Stop
```

### Algorithm : Insert

```
1. Start
2. Read the data to be inserted
3. Index = hkey(data);
4. If a[index].value ==-1
      a. a[index].value=data
5. Else
      a. Repeat I = index + 1 till size of hash table
      b. If a[index].value==-1
            i. a[index].value=data
            ii. Set flag=1
        c. Repeat I =0 till index && flag==0
        d. If a[index].value == -1
            i. a[index].value = data
            ii. flag =0
6. Stop
```

**Algorithm : Search**

```
1. Start
2. Read data to be searched
3. Set index = hkey(data);
4. If a[index].value==data
    a. Print "ELEMENT IS PRESENT"
    b. Return index
    c. Set flag==1
5. If a[index]!=value
    a. Repeat i=index+1 till size of hash table
    b. If a[i].value==data
        i. Print "ELEMENT IS PRESENT";
        ii. Return i
        iii. Set flag=1
    c. Repeat i=0 till index
    d. If a[i].value==data
        i. Print "ELEMENT IS PRESENT"
        ii. Return i
        iii. Set flag=1
6. If flag==0
    a. Print "THE ELEMENT IS ABSENT"
7. Stop
```

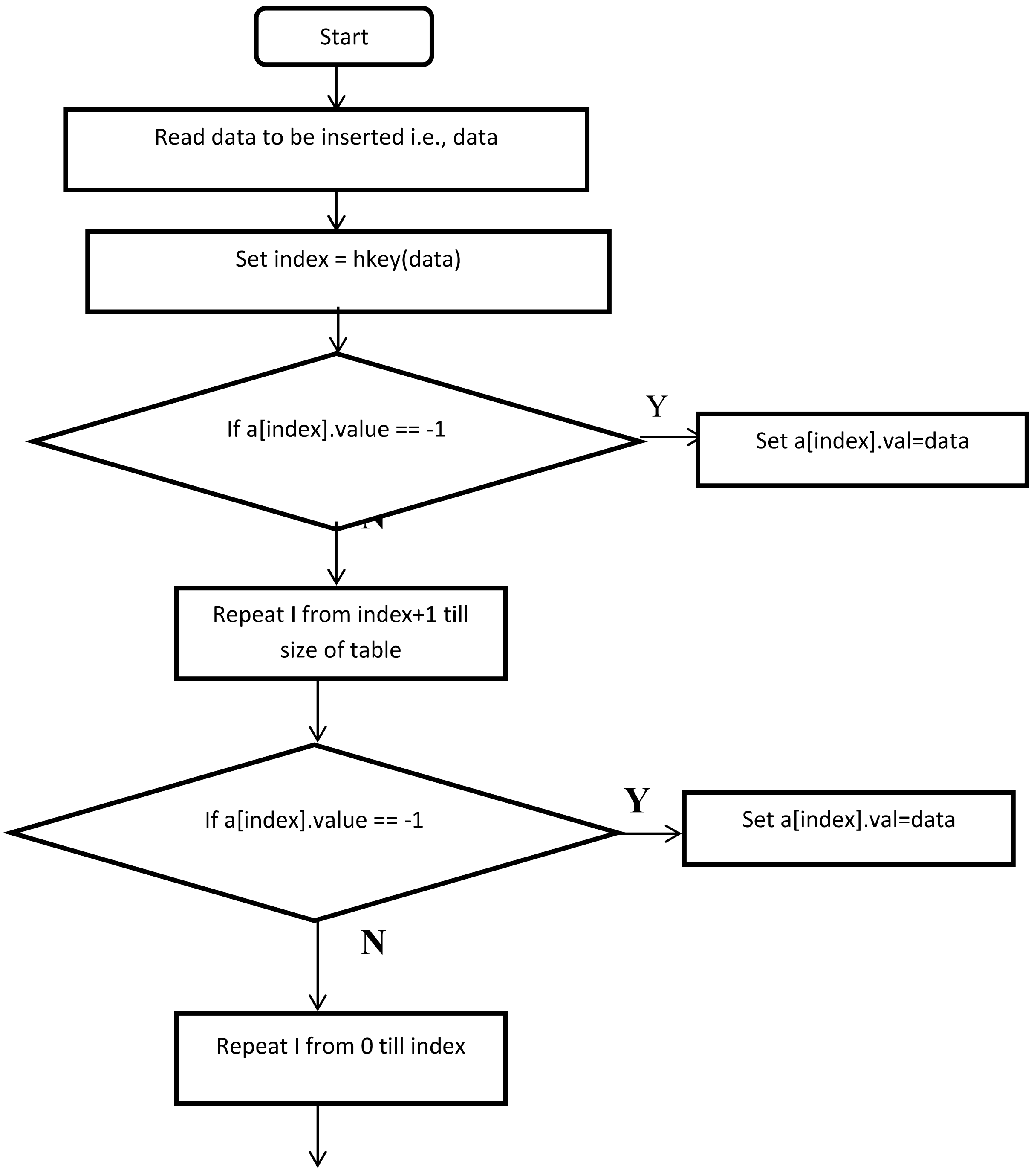
**Algorithm : DELETE RECORD**

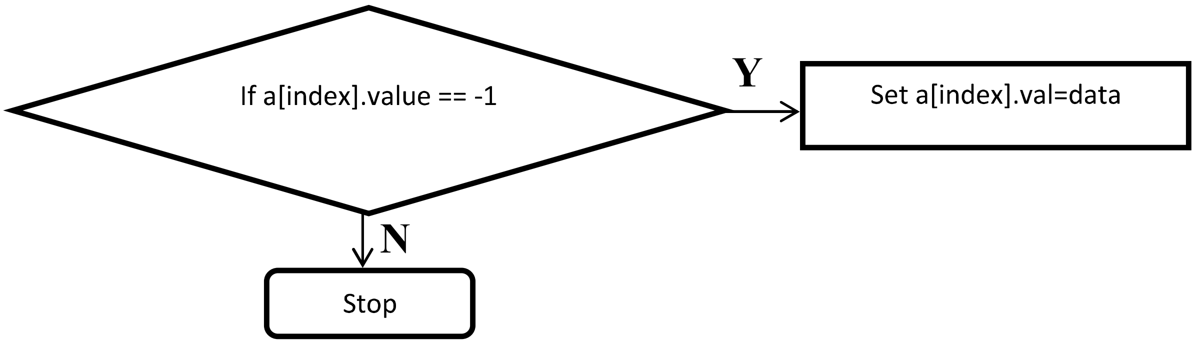
```
1. Start
```

```
2. Read data to be deleted
3. Set index = search(data)
4. If index ! = -99
      a. Set a[index]=-1
5. Stop
```

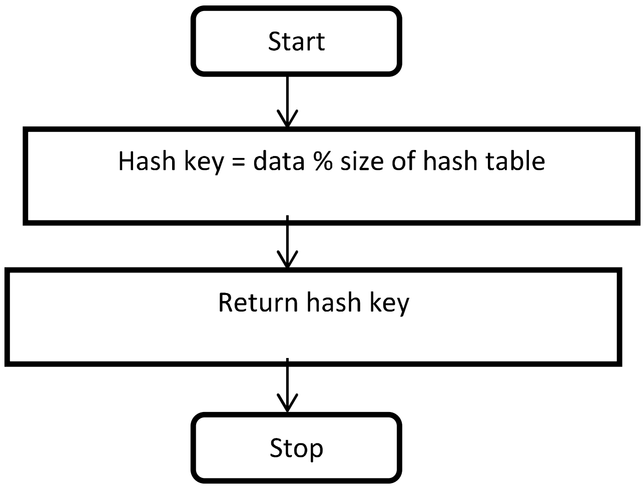
## Flowchart:

## Flowchart: Insert

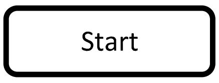


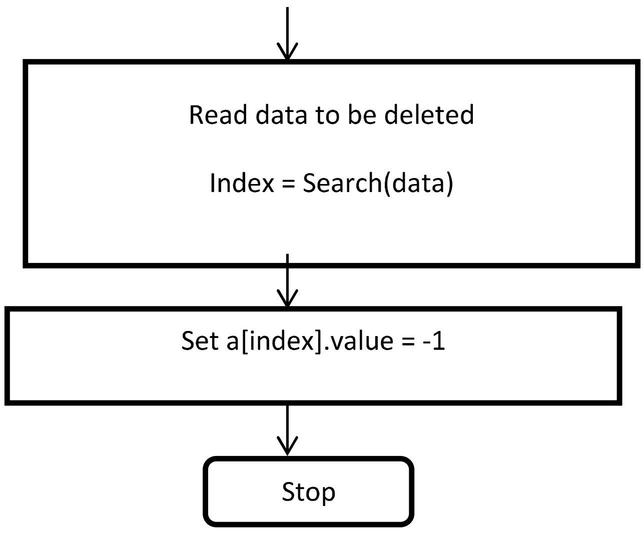


## Flowchart: Hash Key



## Flowchart: Delete





**CONCLUSION :-**

Hence, we have learnt the implementation of hash tables and created a telephone directory and a dictionary.