INDEX

1. Introduction to data structures
2. Array Implementation Of Stack
3. Application Of Stack – Conversion Of Infix To Postfix
4. Implementation Of Linear Queue Using Arrays
5. Array Implementation Of Circular Queue
6. Implementation of Singly Linked List
7. Implementation of Stack using Linked List
8. Implementation of Queue using Linked List
9. Implementation of Bubble Sort
10. Implementation of Insertion Sort
11. Implementation of Quick Sort
12. Implementation of Searching Techniques
13. Implementation of Hashing Technique

Ex. No.: 13

Date :

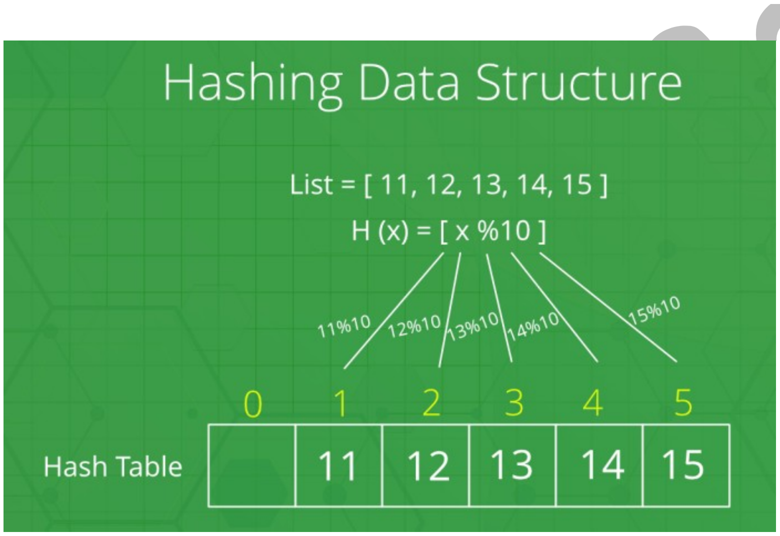
Implementation of Hashing Techniques

Aim: To write a C-program to demonstrate hashing technique.

1. Hash Function:

A **hash function** is any function that can be used to map data of arbitrary size to fixed-size values. The values returned by a hash function are called *hash values*, *hash codes*, *digests*, or simply *hashes*. The values are used to index a fixed-size table called a *hash table*. Use of a hash function to index a hash table is called *hashing* or *scatters storage addressing*.

Hash functions and their associated hash tables are used in data storage and retrieval applications to access data in a small and nearly constant time per retrieval, and storage space only fractionally greater than the total space required for the data or records themselves.



1.1 Algorithm

Algorithm to insert in hash table:

Step1: Get table size from user

Step2: Get input value from user

Step3: send input to hash function to get index

Step4: Insert input to table according to index

Step7: End

1.2 Program

```
#include<conio.h>
#include<stdio.h>
void main()
{
        int a[10]={0,0,0,0,0,0,0,0,0,0};
        int n,value,hashvalue;
        char temp;
//      clrscr();
        printf("Enter the value of n (table size) ::");
        scanf("%d",&n);
        do
        {
                printf("\nEnter the hash value ::");
                scanf("%d",&value);
                hashvalue=value%n;
                if(a[hashvalue]==0)
                {
                        a[hashvalue]=value;
                        printf("\na[%d] The value %d is stored",hashvalue,value);
                }
                else
                {
                        for(hashvalue++;hashvalue<n;hashvalue++)
                        {
                                if(a[hashvalue]==0)
                                {
                    printf("Space is allocated!!!Give another value!!!");
                                        a[hashvalue]=value;
                    printf("\na[%d] The value %d is stored",hashvalue,value);
                                        goto a;
                                }
                        }
                        hashvalue=0;
                        for(hashvalue;hashvalue<n;hashvalue++)
                        {
                                if(a[hashvalue]==0)
                                {
                    printf("Space is allocated!!!Give another value!!!");
                                        a[hashvalue]=value;
                    printf("\na[%d] The value %d is stored",hashvalue,value);
                                        goto a;
                                }
                        }

                }
                a:printf("\nDo you want to enter more? :: ");
                fflush(stdin);
                scanf("%c",&temp);
        }while(temp=='y');
        getch();
}
```