Date: 28/7/25 to 01/08/25

Experiment No: 02

I. Aim: Implement a hash function and use linear probing to handle collisions in hashing.

II. Theory:

Hashing: It is a technique used in data structures that efficiently stores and retrieves data in a way that allows for quick access.

Calculate hash value: The division method involves dividing the key by a prime number and using the remainder as the hash value.

$$h(k)=k \bmod m$$

Collision: When hash function generates same hash value for two different keys then this situation is known as collision.

Collision Resolution Techniques: In open addressing technique all the keys are stored directly into the hash table. When situation arises where two keys are mapped to the same position, the algorithm searches for the next empty slot in the hash table for storing the key.
Linear Probing: In this if a collision occurs, the algorithm searches for the next empty slot in the hash table by moving one position at a time.

III. Program:

```
#include <stdio.h>
#define SIZE 10
int hashTable[SIZE];
void insert(int key)
{
   int index = key%SIZE;
   int i = 0;
   while (hashTable[(index + i) % SIZE] != -1 && i < SIZE)
   {
        i++;
   }
   if (i < SIZE)
        hashTable[(index + i) % SIZE] = key;
   else
        printf("Hash table is full! Cannot insert %d\n", key);
}

void display()
{
   int i;
   printf("\nHash Table:\n");
   for (i = 0; i < SIZE; i++)
   {
        if (hashTable[i] != -1)
          printf("[%d] -> %d\n", i, hashTable[i]);
        else
          printf("[%d] -> Empty\n", i);
   }
}
```

```
void main()
{
   int i,keys[7]={22,32,44,57,68,78,88};
   clrscr();
    for (i = 0; i < SIZE; i++)
   {
         hashTable[i] = -1;
   }
   for (i=0;i <7;i++)
   {
         insert(keys[i]);
   }
   display();
   getch();
 }
```

IV. Output:

V. Complexity Analysis:

Time Complexity:

Initialization executes 10 times – O(10) i.e. O(n)

Inserting Elements- Call to insert function 7 times

Insert function execution:
If no collision then O(1) per key insertion so O(7) i.e. O(n)
If collision occurs then loop executes upto size of an array then O(n) per key.
For 7 (n) keys: 7 * O(n) = n * O(n)=O(n^2)

Display function executes 10 times: O (10) i.e. O(n)

With reference to above:
Best case time complexity is O(n)
Worst case time complexity is O(n^2)

Space Complexity: O(n)

VI. Conclusion: Successfully implemented a hash function and use linear probing to handle collisions in hashing