HASHING (SEPARATE CHAINING)

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 10

struct Node {
   int data;
   struct Node* next;
};

struct Node* hashTable[SIZE];

int hashFunction(int key) {
   return key % SIZE;
}

void insert(int key) {
   int index = hashFunction(key);
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = key;
   newNode->next = hashTable[index];
   hashTable[index] = newNode;
}

void display() {
   for (int i = 0; i < SIZE; i++) {
      printf("%d: ", i);
      struct Node* temp = hashTable[i];
      while (temp) {
         printf("%d -> ", temp->data);
         temp = temp->next;
      }
      printf("NULL\n");
   }
}

int search(int key) {
   int index = hashFunction(key);
   struct Node* temp = hashTable[index];
   while (temp) {
      if (temp->data == key) return 1;
      temp = temp->next;
   }
   return 0;
}

int main() {
   int choice, key;
   while (1) {
      printf("\n1.Insert 2.Display 3.Search 4.Exit\n");
      scanf("%d", &choice);
      if (choice == 1) {
         printf("Enter key: ");
         scanf("%d", &key);
         insert(key);
```

```
    } else if (choice == 2) {
        display();
    } else if (choice == 3) {
        printf("Enter key to search: ");
        scanf("%d", &key);
        if (search(key)) printf("Found\n");
        else printf("Not Found\n");
    } else break;
  }
  return 0;
}
```

OUTPUT: