# Rajalakshmi Engineering College

Name: Parameswari P
Email: 240701378@rajalakshmi.edu.in
Roll no: 240701378
Phone: 9500133836
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

Scan to verify results

## NeoColab_REC_CS23231_DATA STRUCTURES

## REC_DS using C_Week 7_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

## Section 1 : Coding

1. Problem Statement

In a messaging application, users maintain a contact list with names and corresponding phone numbers. Develop a program to manage this contact list using a dictionary implemented with hashing.

The program allows users to add contacts, delete contacts, and check if a specific contact exists. Additionally, it provides an option to print the contact list in the order of insertion.

*Input Format*

The first line consists of an integer n, representing the number of contact pairs to be inserted.

Each of the next n lines consists of two strings separated by a space: the name of the contact (key) and the corresponding phone number (value).

The last line contains a string k, representing the contact to be checked or removed.

***Output Format***

If the given contact exists in the dictionary:

1. The first line prints "The given key is removed!" after removing it.
2. The next n - 1 lines print the updated contact list in the format: "Key: X; Value: Y" where X represents the contact's name and Y represents the phone number.

If the given contact does not exist in the dictionary:

1. The first line prints "The given key is not found!".
2. The next n lines print the original contact list in the format: "Key: X; Value: Y" where X represents the contact's name and Y represents the phone number.

Refer to the sample outputs for the formatting specifications.

***Sample Test Case***

```
Input: 3
Alice 1234567890
Bob 9876543210
Charlie 4567890123
Bob
```

```
Output: The given key is removed!
Key: Alice; Value: 1234567890
Key: Charlie; Value: 4567890123
```

***Answer***

```
// You are using GCC
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#define MAX 50
```

```
#define SIZE 101

typedef struct Contact {
   char name[20];
   char phone[20];
   int active;
} Contact;

Contact *hash_table[SIZE];
Contact ordered_list[MAX];
int count = 0;


int hash(char *key) {
   int sum = 0;
   for (int i = 0; key[i]; i++) {
      sum += key[i];
   }
   return sum % SIZE;
}


void insert_contact(char *name, char *phone) {
   // Save in ordered list
   strcpy(ordered_list[count].name, name);
   strcpy(ordered_list[count].phone, phone);
   ordered_list[count].active = 1;
   count++;
   int idx = hash(name);
   while (hash_table[idx] != NULL) {
      idx = (idx + 1) % SIZE;
   }

   hash_table[idx] = (Contact *)malloc(sizeof(Contact));
   strcpy(hash_table[idx]->name, name);
   strcpy(hash_table[idx]->phone, phone);
   hash_table[idx]->active = 1;
}
int search_contact(char *key, int *index) {
   int idx = hash(key);
   int start = idx;
```

```
    while (hash_table[idx] != NULL) {
        if (strcmp(hash_table[idx]->name, key) == 0 && hash_table[idx]->active) {
            *index = idx;
            return 1;
        }
        idx = (idx + 1) % SIZE;
        if (idx == start) break;
    }
    return 0;
}

void delete_contact(char *key) {
    int idx;
    if (search_contact(key, &idx)) {
        hash_table[idx]->active = 0;
        for (int i = 0; i < count; i++) {
            if (strcmp(ordered_list[i].name, key) == 0 && ordered_list[i].active) {
                ordered_list[i].active = 0;
                break;
            }
        }

        printf("The given key is removed!\n");
        for (int i = 0; i < count; i++) {
            if (ordered_list[i].active) {
                printf("Key: %s; Value: %s\n", ordered_list[i].name, ordered_list[i].phone);
            }
        }
    } else {
        printf("The given key is not found!\n");
        for (int i = 0; i < count; i++) {
            if (ordered_list[i].active) {
                printf("Key: %s; Value: %s\n", ordered_list[i].name, ordered_list[i].phone);
            }
        }
    }
}

int main() {
    int n;
    scanf("%d", &n);
    getchar();
    char name[20], phone[20];

    for (int i = 0; i < n; i++) {
        scanf("%s %s", name, phone);
        insert_contact(name, phone);
    }
    char key[20];
    scanf("%s", key);
    delete_contact(key);
    for (int i = 0; i < SIZE; i++) {
        if (hash_table[i] != NULL) {
            free(hash_table[i]);
        }
    }

    return 0;
}
```

***Status :*** Correct ***Marks : 10/10***