

Rajalakshmi Engineering College

Name: paranidharan R
Email: 240801238@rajalakshmi.edu.in
Roll no: 240801238
Phone: 9360861582
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

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 <stdlib.h>
```

```
#include <string.h>
```

```
#define MAX_CONTACTS 50
```

```
#define MAX_NAME_LENGTH 11
#define MAX_PHONE_LENGTH 11
```

```
typedef struct {
    char name[MAX_NAME_LENGTH];
    char phone[MAX_PHONE_LENGTH];
} Contact;
```

```
int find_contact(Contact contacts[], int n, char *key) {
    for (int i = 0; i < n; i++) {
        if (strcmp(contacts[i].name, key) == 0) {
            return i;
        }
    }
    return -1;
}
```

```
int main() {
    int n;
    scanf("%d", &n);
```

```
    Contact contacts[MAX_CONTACTS];
```

```
    for (int i = 0; i < n; i++) {
        scanf("%s %s", contacts[i].name, contacts[i].phone);
    }
```

```
    char key[MAX_NAME_LENGTH];
    scanf("%s", key);
```

```
    int index = find_contact(contacts, n, key);
```

```
    if (index != -1) {
        printf("The given key is removed!\n");
        for (int i = index; i < n - 1; i++) {
            contacts[i] = contacts[i + 1];
        }
        n--;
    } else {
        printf("The given key is not found!\n");
    }
}
```

```
for (int i = 0; i < n; i++) {  
    printf("Key: %s; Value: %s\n", contacts[i].name, contacts[i].phone);  
}  
  
return 0;  
}
```

Status : Correct

Marks : 10/10