Q) Consider the telephone book database of ‘n’ clients. Make use of a hash table implementation to quickly look up a client’s telephone number. Make use of Linear Probing, Double Hashing and Quadratic Collision hashing techniques.

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

#define MAX_CLIENTS 10
typedef struct {
    char name[50];
    char phoneNumber[15];
    int isOccupied;  // 0 = empty, 1 = occupied, -1 = deleted
} Client;
Client hashTable[MAX_CLIENTS];
int hashFunction(char *name) {
    int hashValue = 0;
    for (int i = 0; name[i] != '\0'; i++) {
        hashValue += (int)name[i]; // Sum of ASCII values of each character
    }
    return hashValue % MAX_CLIENTS;
}
int linearProbing(int index) {
    return (index + 1) % MAX_CLIENTS;
}
int doubleHashing(int index, int i) {
    int secondaryHash = 1 + (index % (MAX_CLIENTS - 1)); // Example secondary hash function
    return (index + i * secondaryHash) % MAX_CLIENTS;
}
int quadraticProbing(int index, int i) {
    return (index + i * i) % MAX_CLIENTS;
}
```

```
void insertClient(char *name, char *phoneNumber, int collisionMethod) {
    int index = hashFunction(name);
    int i = 0;
    while (hashTable[index].isOccupied == 1) {
        if (strcmp(hashTable[index].name, name) == 0) {
            printf("Client already exists.\n");
            return;
        }
        switch (collisionMethod) {
            case 1: // Linear Probing
                index = linearProbing(index);
                break;
            case 2: // Double Hashing
                index = doubleHashing(index, i);
                break;
            case 3: // Quadratic Probing
                index = quadraticProbing(index, i);
                break;
            default:
                printf("Invalid collision method.\n");
                return;
        }
        i++;
    }
    strcpy(hashTable[index].name, name);
    strcpy(hashTable[index].phoneNumber, phoneNumber);
    hashTable[index].isOccupied = 1;

    printf("Client inserted successfully.\n");
}
void searchClient(char *name) {
```

```
    int index = hashFunction(name);
    int i = 0;

    while (hashTable[index].isOccupied != 0) {
        if (hashTable[index].isOccupied == 1 && strcmp(hashTable[index].name, name) == 0) {
            printf("Client Found: %s, Phone Number: %s\n", hashTable[index].name,
hashTable[index].phoneNumber);
            return;
        }
        index = linearProbing(index);
        i++;
    }
 printf("Client not found.\n");
}
void displayClients() {
    printf("\nPhonebook:\n");
    for (int i = 0; i < MAX_CLIENTS; i++) {
        if (hashTable[i].isOccupied == 1) {
            printf("Name: %s, Phone Number: %s\n", hashTable[i].name, hashTable[i].phoneNumber);
        }
    }
}
void menu() {
    int choice, collisionMethod;
    char name[50], phoneNumber[15];

    do {
        printf("\nTelephone Book Database\n");
        printf("1. Insert Client\n");
        printf("2. Search Client\n");
        printf("3. Display Clients\n");
```

```
printf("4. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
getchar(); // Clear newline character

switch (choice) {
   case 1:
      printf("Enter client name: ");
      fgets(name, 50, stdin);
      name[strcspn(name, "\n")] = 0; // Remove newline

      printf("Enter phone number: ");
      fgets(phoneNumber, 15, stdin);
      phoneNumber[strcspn(phoneNumber, "\n")] = 0;

      printf("\nChoose collision handling method:\n");
      printf("1. Linear Probing\n");
      printf("2. Double Hashing\n");
      printf("3. Quadratic Probing\n");
      printf("Enter your choice: ");
      scanf("%d", &collisionMethod);

      insertClient(name, phoneNumber, collisionMethod);
      break;

   case 2:
      printf("Enter client name to search: ");
      fgets(name, 50, stdin);
      name[strcspn(name, "\n")] = 0;
      searchClient(name);
      break;
```

```
            case 3:
                displayClients();
                break;

            case 4:
                printf("Exiting program.\n");
                break;

            default:
                printf("Invalid choice, try again.\n");
        }
    } while (choice != 4);
}

int main() {
    for (int i = 0; i < MAX_CLIENTS; i++) {
        hashTable[i].isOccupied = 0;
    }
    menu();
    return 0;
}
```

Output Clear

```
Telephone Book Database
1. Insert Client
2. Search Client
3. Display Clients
4. Exit
Enter your choice: 1
Enter client name: Jonh
Enter phone number: 9999970052

Choose collision handling method:
1. Linear Probing
2. Double Hashing
3. Quadratic Probing
Enter your choice: 2
Client inserted successfully.
```

```
Telephone Book Database
1. Insert Client
2. Search Client
3. Display Clients
4. Exit
Enter your choice: 3

Phonebook:
Name: Jonh , Phone Number: 9999970052

Telephone Book Database
1. Insert Client
2. Search Client
3. Display Clients
4. Exit
Enter your choice:
=== Session Ended. Please Run the code again ===
```