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

int hashTable[SIZE];

// Initialize hash table with -1 (empty slot)
void initialize() {
    for (int i = 0; i < SIZE; i++)
        hashTable[i] = -1;
}

// Hash function
int hash(int key) {
    return key % SIZE;
}

// Insert using linear probing
void insert(int key) {
    int index = hash(key);
    int start = index;

    // Linear probing
    while (hashTable[index] != -1) {
        index = (index + 1) % SIZE;
        if (index == start) {
            printf("Hash table is full! Cannot insert key %d\n", key);
            return;
        }
    }
    hashTable[index] = key;
}
```

```
// Search for a key
int search(int key) {
    int index = hash(key);
    int start = index;

    while (hashTable[index] != -1) {
        if (hashTable[index] == key)
            return index;
        index = (index + 1) % SIZE;
        if (index == start)
            break;
    }
    return -1; // Not found
}

// Display hash table
void display() {
    printf("Hash Table:\n");
    for (int i = 0; i < SIZE; i++) {
        if (hashTable[i] != -1)
            printf("[%d] --> %d\n", i, hashTable[i]);
        else
            printf("[%d] --> EMPTY\n", i);
    }
}

int main() {
    int n, key, choice;

    initialize();
```

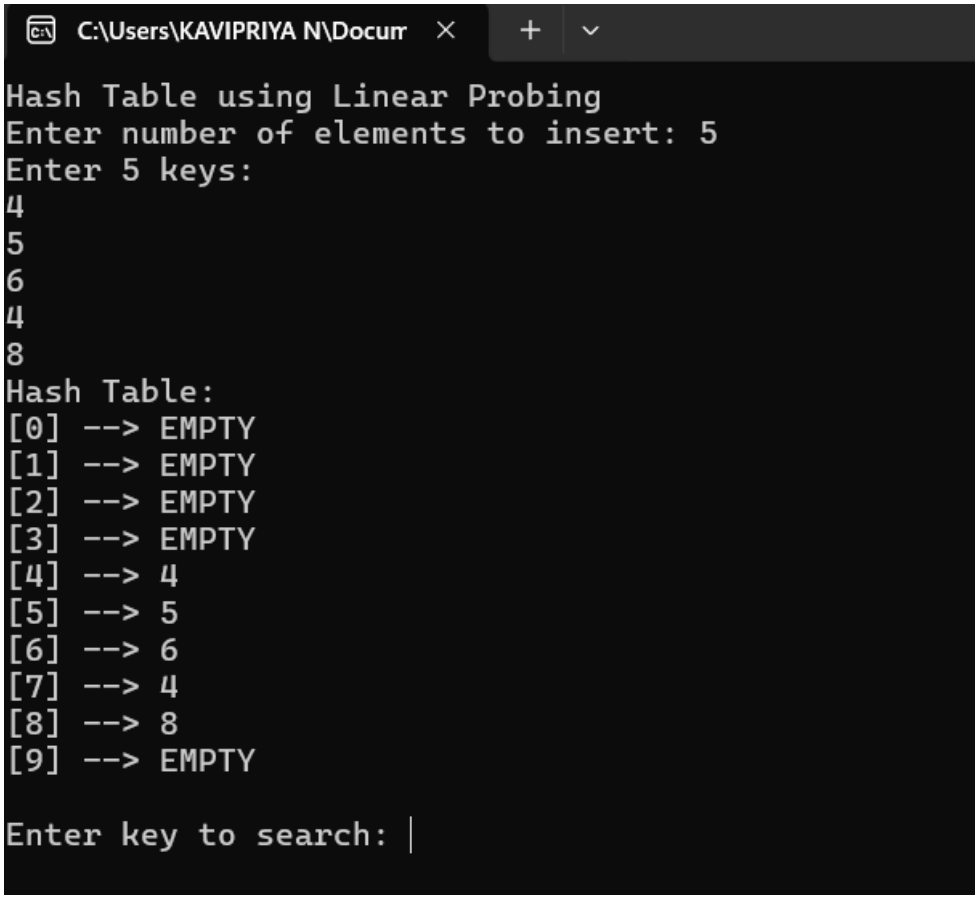
```
    printf("Hash Table using Linear Probing\n");
    printf("Enter number of elements to insert: ");
    scanf("%d", &n);

    printf("Enter %d keys:\n", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &key);
        insert(key);
    }

    display();

    printf("\nEnter key to search: ");
    scanf("%d", &key);
    int index = search(key);
    if (index != -1)
        printf("Key %d found at index %d\n", key, index);
    else
        printf("Key %d not found in the hash table\n", key);

    return 0;
}
```


C:\Users\KAVIPRIYA N\Docum
Hash Table using Linear Probing
Enter number of elements to insert: 5
Enter 5 keys:
4
5
6
4
8
Hash Table:
[0] --> EMPTY
[1] --> EMPTY
[2] --> EMPTY
[3] --> EMPTY
[4] --> 4
[5] --> 5
[6] --> 6
[7] --> 4
[8] --> 8
[9] --> EMPTY
Enter key to search: