#include <stdio.h>

#include <stdlib.h>

#define TABLE\_SIZE 10 // Define the size of the hash table

int hashTable[TABLE\_SIZE]; // Hash table

int currentSize = 0; // To track the number of elements in the table

// Initialize the hash table

void initializeTable() {

for (int i = 0; i < TABLE\_SIZE; i++) {

hashTable[i] = -1; // -1 indicates an empty slot

}

}

// Hash function

int hashFunction(int key) {

return key % TABLE\_SIZE;

}

// Insert a key into the hash table

void insert(int key) {

if (currentSize >= TABLE\_SIZE) {

printf("Hash table is full!\n");

return;

}

int index = hashFunction(key);

int startIndex = index; // Save the starting index for full traversal detection

// Linear probing to find an empty slot

while (hashTable[index] != -1) {

index = (index + 1) % TABLE\_SIZE;

if (index == startIndex) {

printf("No empty slot found for key %d\n", key);

return;

}

}

hashTable[index] = key;

currentSize++;

printf("Inserted %d at index %d\n", key, index);

}

// Search for a key in the hash table

int search(int key) {

int index = hashFunction(key);

int startIndex = index;

while (hashTable[index] != -1) {

if (hashTable[index] == key) {

return index; // Key found

}

index = (index + 1) % TABLE\_SIZE;

// If we've looped back to the starting index, stop

if (index == startIndex) {

break;

}

}

return -1; // Key not found

}

// Delete a key from the hash table

void delete (int key) {

int index = search(key);

if (index == -1) {

printf("Key %d not found in the hash table.\n", key);

return;

}

// Mark the slot as deleted by setting it to -1

hashTable[index] = -1;

currentSize--;

printf("Key %d deleted from index %d\n", key, index);

}

// Display the hash table

void display() {

printf("Hash Table:\n");

for (int i = 0; i < TABLE\_SIZE; i++) {

if (hashTable[i] == -1) {

printf("[%d] - Empty\n", i);

} else {

printf("[%d] - %d\n", i, hashTable[i]);

}

}

}

int main() {

initializeTable();

int choice, key;

do {

printf("\nHashing with Linear Probing:\n");

printf("1. Insert\n2. Search\n3. Delete\n4. Display\n5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter key to insert: ");

scanf("%d", &key);

insert(key);

break;

case 2:

printf("Enter key to search: ");

scanf("%d", &key);

int result = search(key);

if (result == -1) {

printf("Key %d not found in the hash table.\n", key);

} else {

printf("Key %d found at index %d.\n", key, result);

}

break;

case 3:

printf("Enter key to delete: ");

scanf("%d", &key);

delete (key);

break;

case 4:

display();

break;

case 5:

printf("Exiting program.\n");

break;

default:

printf("Invalid choice. Please try again.\n");

}

} while (choice != 5);

return 0;

}