ASSIGNMENT 12

Name: Aniket Kalbhor

Roll: 48

PRN: 12210601

Linear Probing

#include <stdio.h>

#include <stdlib.h>

#define TABLE\_SIZE 10

int hashTable[TABLE\_SIZE] = {0};

void displayHashTable() {

    printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

    printf("Hash Table:\n");

    printf("Index\tValue\n");

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

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

}

void insert()

{

    int key, index, i, flag = 0, hashkey;

    printf("Enter value to be inserted in hash table: ");

    scanf("%d",&key);

    hashkey = key%TABLE\_SIZE;

    for(i = 0;i < TABLE\_SIZE;i++)

    {

        index = (hashkey+i)%TABLE\_SIZE;

        if (hashTable[index] == 0)

        {

            hashTable[index] = key;

            printf("Element inserted at index %d\n", index);

            break;

        }

    }

    if (i == TABLE\_SIZE)

    {

        printf("Element can't be inserted\n");

    }

}

int main() {

    int choice;

    do {

        printf("\n1. Insert\t");

        printf("2. Display\t");

        printf("3. Exit\n");

        printf("Enter your choice:");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                insert();

                break;

            case 2:

                displayHashTable();

                break;

            case 3:

                printf("Exiting...\n");

                break;

            default:

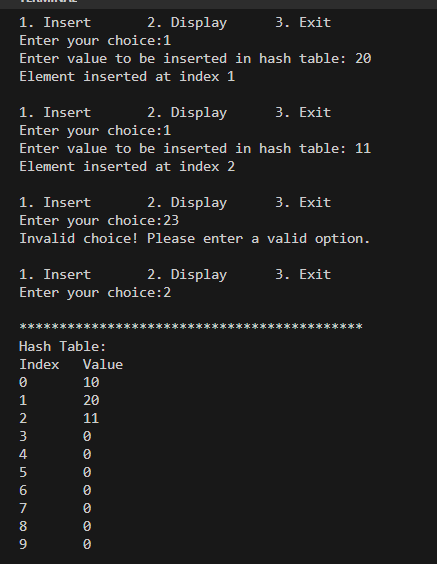
                printf("Invalid choice! Please enter a valid option.\n");

        }

    } while (choice != 3);

    return 0;

}



Quadratic probing

#include <stdio.h>

#include <stdlib.h>

#define TABLE\_SIZE 10

int hashTable[TABLE\_SIZE] = {0};

int hash(int key) {

    return key % TABLE\_SIZE;

}

void displayHashTable() {

    printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

    printf("Hash Table:\n");

    printf("Index\tValue\n");

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

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

}

void insert(int key) {

    int index, step, i;

    index = hash(key);

    step = 1;

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

        if (hashTable[index] == 0) {

            hashTable[index] = key;

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

            return;

        }

        index = (index + step \* step) % TABLE\_SIZE;

        step++;

    }

    printf("Hash table is full. Element %d cannot be inserted.\n", key);

}

int main() {

    int choice, key;

    do {

        printf("\n1. Insert\t");

        printf("2. Display\t");

        printf("3. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Enter value to be inserted in hash table: ");

                scanf("%d", &key);

                insert(key);

                break;

            case 2:

                displayHashTable();

                break;

            case 3:

                printf("Exiting...\n");

                break;

            default:

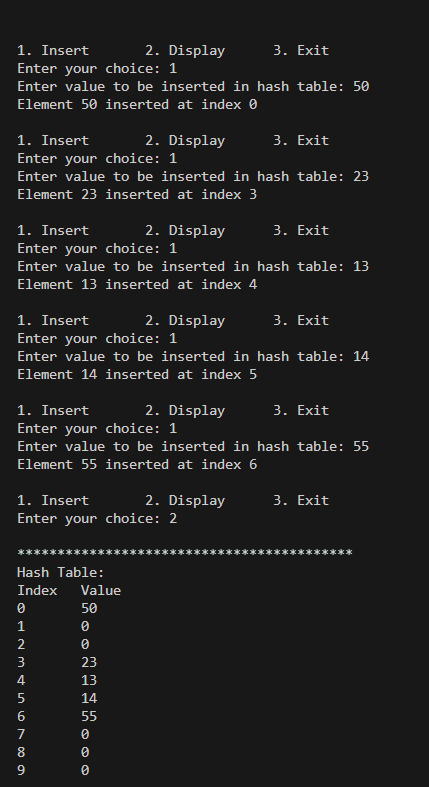
                printf("Invalid choice! Please enter a valid option.\n");

        }

    } while (choice != 3);

    return 0;

}



Double Hashing

#include <stdio.h>

#include <stdlib.h>

#define TABLE\_SIZE 10

int hashTable[TABLE\_SIZE] = {0};

void displayHashTable() {

    printf("Hash Table:\n");

    printf("Index\tValue\n");

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

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

    }

}

int hash1(int key) {

    return key % TABLE\_SIZE;

}

int hash2(int key) {

    // Choose a prime smaller than TABLE\_SIZE for best results

    return 7 - (key % 7);

}

void insert(int key) {

    int index, step, i;

    index = hash1(key);

    step = hash2(key);

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

        if (hashTable[index] == 0) {

            hashTable[index] = key;

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

            return;

        }

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

    }

    printf("Hash table is full. Element %d cannot be inserted.\n", key);

}

int main() {

    int choice, key;

    do {

        printf("\n1. Insert\t");

        printf("2. Display\t");

        printf("3. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Enter value to be inserted in hash table: ");

                scanf("%d", &key);

                insert(key);

                break;

            case 2:

                displayHashTable();

                break;

            case 3:

                printf("Exiting...\n");

                break;

            default:

                printf("Invalid choice! Please enter a valid option.\n");

        }

    } while (choice != 3);

    return 0;

}

