**Data Structures Lab [BCSL305]**

**Program 12**

**Given a File of N employee records with a set K of Keys(4 -digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table(HT) of m memory locations with L as the set of memory addresses (2- digit) of locations in HT. Let the keys in K and addresses in L are Integers. Design and develop a Program in C that uses Hash function H: K → L as H(K)=K mod m (remainder method)**

#include <stdio.h>

#include <stdlib.h>

#define MAX\_EMPLOYEES 50

#define MAX\_MEMORY\_LOCATIONS 100

// Employee record structure

struct Employee {

int key; // Unique key (4-digit)

char name[50]; // Employee's name

int age; // Employee's age

};

// Hash Table (HT) to store the employee records

struct Employee\* hashTable[MAX\_MEMORY\_LOCATIONS];

// Hash Function: H(K) = K mod m

int hashFunction(int key, int m) {

return key % m;

}

// Function to insert an employee record into the hash table

void insertEmployee(int key, char\* name, int age, int m) {

int hashIndex = hashFunction(key, m);

// Linear probing for collision handling

while (hashTable[hashIndex] != NULL) {

hashIndex = (hashIndex + 1) % m;

}

// Create a new employee record and store it in the hash table

struct Employee\* emp = (struct Employee\*)malloc(sizeof(struct Employee));

emp->key = key;

snprintf(emp->name, sizeof(emp->name), "%s", name);

emp->age = age;

hashTable[hashIndex] = emp;

}

// Function to display the employee records in the hash table

void displayHashTable(int m) {

printf("\nEmployee Records in the Hash Table:\n");

for (int i = 0; i < m; i++) {

if (hashTable[i] != NULL) {

printf("Memory Location %d -> Key: %d, Name: %s, Age: %d\n", i, hashTable[i]->key, hashTable[i]->name, hashTable[i]->age);

}

}

}

// Function to search for an employee by key

void searchEmployee(int key, int m) {

int hashIndex = hashFunction(key, m);

// Linear probing for collision handling

while (hashTable[hashIndex] != NULL) {

if (hashTable[hashIndex]->key == key) {

printf("\nEmployee Found: Key: %d, Name: %s, Age: %d\n", hashTable[hashIndex]->key, hashTable[hashIndex]->name, hashTable[hashIndex]->age);

return;

}

hashIndex = (hashIndex + 1) % m;

}

printf("\nEmployee with Key %d not found.\n", key);

}

int main() {

int m, n, key, age;

char name[50];

// Initialize hash table

for (int i = 0; i < MAX\_MEMORY\_LOCATIONS; i++) {

hashTable[i] = NULL;

}

printf("Enter the number of memory locations (m) in the hash table: ");

scanf("%d", &m);

printf("Enter the number of employees (N): ");

scanf("%d", &n);

// Input employee records and insert them into the hash table

for (int i = 0; i < n; i++) {

printf("\nEnter employee %d details:\n", i + 1);

printf("Key (4-digit): ");

scanf("%d", &key);

printf("Name: ");

scanf("%s", name);

printf("Age: ");

scanf("%d", &age);

insertEmployee(key, name, age, m);

}

// Display the hash table

displayHashTable(m);

// Search for an employee by key

printf("\nEnter a key to search for an employee: ");

scanf("%d", &key);

searchEmployee(key, m);

return 0;

}