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

int empkey[20], hashindex, n, m, *ht, elecount = 0;

void createhashtable() {
    ht = (int*)malloc(m * sizeof(int));
    if (ht == NULL) {
        printf("MEMORY UNAVAILABLE!");
    } else {
        for (int i = 0; i < m; i++) {
            ht[i] = -1;
        }
    }
}

void insertintohashtable(int key) {
    hashindex = key % m;
    printf("Key %d hashed to index %d.\n", key, hashindex);

    // Linear probing to find the next available slot
    while (ht[hashindex] != -1) {
        printf("Collision at index %d. Probing next index.\n", hashindex);
        hashindex = (hashindex + 1) % m;
    }

    ht[hashindex] = key;
    printf("Key %d inserted at index %d.\n", key, hashindex);
    elecount++;
}


void display() {
    int i;
    if (elecount == 0) {
        printf("HASH TABLE EMPTY\n");
    }
    for (i = 0; i < m; i++) {
        printf("T[%d] --> %d\n", i, ht[i]);
    }
}

void main() {
    int i;
    printf("Enter no of records: ");
    scanf("%d", &n);
    printf("Enter hashtable size: ");
    scanf("%d", &m);
    printf("Enter emp key values: \n");
    for (i = 0; i < n; i++) {
        scanf("%d", &empkey[i]);
    }

    createhashtable();
    printf("Inserting keys into hashtable:\n");
    for (i = 0; i < n; i++) {
        if (elecount == m) {
            printf("HASH TABLE FULL\n");
            printf("Can't insert %d key\n", empkey[i]);
            break;
        }
        insertintohashtable(empkey[i]);
    }

    display();
}

```