S.No: 12	Exp. Name: ***Hash Table***	Date:2025-10-18

Aim:

You are entrusted with the task of developing a program in C to manage employee records efficiently. The program operates on a set of **N** employee records, each uniquely identified by a four-digit key.

The program relies on a Hash Table (HT) that is responsible for maintaining these employee records in memory. The Hash Table comprises **m** memory locations, and the set **L** represents the memory addresses, each a two-digit value. Each of these addresses corresponds to one of the four locations within the Hash Table.

Resolve the collision (if any) using **linear probing**.

Input Format:
First line of input contains two space separated integers N, m respectively
Second line of input contains N space separated integer representing the four digit key values

Output Format:
Print the hash table constructed as per the test cases given

Source Code:

HashTable1.c

```
// Type your content here...
#include <stdio.h>
#include<stdlib.h>
int main(){
   int n, m;
   if(scanf("%d %d", &n, &m)!=2) return 0;
   int *keys = NULL;

   if(n>0){
      keys = (int *)malloc(sizeof(int)*n);
      for(int i = 0; i<n; i++)
         scanf("%d", &keys[i]);
   }

   int *hash = (int *)malloc(sizeof(int)*m);
   for(int i = 0; i<m; i++)
      hash[i]=-1;
   int fail = -1;
   for(int k = 0; k<n;k++){
      int key = keys[k];
      int index = key%m;
      int inserted = 0;
      int pos;
      int probe = 0;
      while(probe<m){
         pos = (index+probe)%m;
         if(hash[pos]==-1){
            hash[pos]=key;
            inserted=1;
            break;
         }
```

```
            probe++;
        }
        if(!inserted){
            fail=k+1;
            break;
        }
    }
    if(fail!=-1){
        printf("Hash table is full. Cannot insert records from key %d\n", fail);
    }
    int empty = 1;
    for(int i = 0; i<m; i++){
        if(hash[i]!=-1){
            empty = 0;
            break;
        }
    }
    if(empty)
        printf("Hash Table is empty\n");
    else{
        for(int i = 0; i<m; ++i){
            if(hash[i]==-1)
                printf("T[%d] -> -1\n", i);
            else
                printf("T[%d] -> %d\n", i, hash[i]);
        }
    }
    free(keys);
    free(hash);
    return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
5 5
1596 3574 9512 7532 4268
T[0] -> 4268
T[1] -> 1596
T[2] -> 9512
T[3] -> 7532
T[4] -> 3574

Test Case - 2
User Output
6 4
4587 6589 4521 6523 5824 6548
Hash table is full. Cannot insert records from key 5
T[0] -> 6523
T[1] -> 6589
T[2] -> 4521

```
T[3] -> 4587
```

Test Case - 3
User Output
0 5
Hash Table is empty