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), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.

```
#include<stdio.h>

#include<stdlib.h>

int key[20], n, m;
int * ht, index;
int count = 0;

void insert(int key)
{
   index = key % m;
   while (ht[index] != -1)
   {
      index = (index + 1) % m;
   }
   ht[index] = key;
   count++;
}

void display()
{
   int i;
   if (count == 0)
   {
      printf("\nHash Table is empty");
      return;
   }

   printf("\nHash Table contents are:\n ");
   for (i = 0; i < m; i++)
      printf("\n T[%d] --> %d ", i, ht[i]);
}

void main()
```

```
{
   int i;
   printf("\nEnter the number of employee  records (N): ");
   scanf("%d", & n);

   printf("\nEnter the two digit memory locations (m) for hash table: ");
   scanf("%d", & m);

   ht = (int * ) malloc(m * sizeof(int));
   for (i = 0; i < m; i++)
      ht[i] = -1;

   printf("\nEnter the four digit key values (K) for N Employee Records:\n ");
   for (i = 0; i < n; i++)
      scanf("%d", & key[i]);

   for (i = 0; i < n; i++)
   {
      if (count == m)
      {
         printf("\n-----Hash table is full. Cannot insert the record %d key-----", i + 1);
         break;
      }
      insert(key[i]);
   }

   display();
}
```

OUTPUT

```
Enter the number of employee  records (N) :10
Enter the two digit memory locations (m) for hash table:15
Enter the four digit key values (K) for N Employee Records:
 4020
 4560
 9908
 6785
 0423
 7890
 6547
 3342
 9043
 6754
 Hash Table contents are:

 T[0] --> 4020
 T[1] --> 4560
```

T[2] --> 7890
T[3] --> 423
T[4] --> 6754
T[5] --> 6785
T[6] --> -1
T[7] --> 6547
T[8] --> 9908
T[9] --> -1
T[10] --> -1
T[11] --> -1
T[12] --> 3342
T[13] --> 9043
T[14] --> -1