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. 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>

#define MAX 10

struct employee

{

         int id;

        char name[15];

};

typedef struct employee EMP;

EMP emp[MAX];

int a[MAX];

int create(int num)

{

        int key;

        key = num % 100;

        return key;

}

int getemp(EMP emp[],int key)

{

        printf("\nEnter emp id: ");

        scanf("%d",&emp[key].id);

        printf("\nEnter emp name: ");

        scanf("%s",emp[key].name);

        return key;
```

```
}
void display()
{
        int i, ch;
        printf("\n1.Display ALL\n2.Filtered Display");
        printf("\nEnter the choice: ");
        scanf("%d",&ch);
        if(ch == 1)
        {
                printf("\nThe hash table is:\n");
                printf("\nHTKey\tEmpID\tEmpName");
                for(i=0; i<MAX; i++)
                printf("\n%d\t%d\t%s", i, emp[i].id, emp[i].name);
        }
        else
        {
                printf("\nThe hash table is:\n");
                printf("\nHTKey\tEmpID\tEmpName");
                for(i=0; i<MAX; i++)
                if(a[i] != -1)
                {
                        printf("\n%d\t%d\t%s", i, emp[i].id, emp[i].name);
                        continue;
                }
        }
}
void linear_prob(int key, int num)
{
        int flag, i, count = 0;
        flag = 0;
```

```
if(a[key] == -1)

{

        a[key]=getemp(emp, key);

}

else

{

        printf("\nCollision Detected...!!!\n");

        i = 0;

        while(i < MAX)

        {

                if (a[i] != -1)
                {
                    count++;
                    break;
                }
                else
                i++;
        }

        printf("\nCollision avoided successfully using LINEAR PROBING\n");

        if(count == MAX)

        {

                printf("\n Hash table is full");

                display(emp);

                exit(1);

        }

        else

        {

                getemp(emp,key+1);

        }

                for(i=key; i<MAX; i++)

                if(a[i] == -1)

                {

                        a[i] = num;
                flag = 1;
                break;
            }
            i = 0;
        while((i < key) && (flag == 0))
        {
            if(a[i] == -1)
            {
            a[i] = num;
            flag=1;
            break;
        }
        i++;
    } // end while
    } // end else
    } // end linear_prob()
void main()
{
    int num, key, i;
    int ans = 1;
    printf("\nCollision handling by linear probing: ");
    for (i=0; i < MAX; i++)
    {
        a[i] = -1;
    }
    do
    {
        printf("\nEnter the data: ");
        scanf("%d", &num);
        key=create(num);
        linear_prob(key,num);
        printf("\nDo you wish to continue? (1/0): ");
        scanf("%d",&ans);
    }while(ans);
    display(emp);
}
```