Program 13: hashing

pseudocode:
```
void insert(int key){
    int index = key % m;
    int startindex = index;
    while(hashTable[index] != -1){
        index = (index+1) % m;
        if (index == start Index){
            printf("Table is full");
            return;
    }}
```

code:
```
#include<stdio.h>

#define MAX 100

int hashTable[MAX];
int m;

void insert(int key){
    int index = key % m;
    int startIndex = index;

    while (hashTable[inda] != -1){
        index = (index+1) % m;

        if (index == startIndex){
            printf("Hash Table is full");
            return ~~0~~;
        }
    }
    hashTable[index] = key;
}
```

```
void display(){
   printf(" HASH TABLE");
   for(int i=0; i<m; i++){
     if(hashTable[i] == -1)
        printf(" HT[%d] --> EMPTY ", i);
     else
        printf(" HT[%d] --> %d\n", i, hashTable[i]);
   }
}


int main(){
    int n, key;

    printf("Enter size of Hash Table : ");
    scanf("%d", &m);

    for(int i=0; i<m; i++)
         hashTable[i] = -1;

    printf(" Enter number of employee records :");
    scanf("%d", &n);

    printf(" Enter %d employee keys (4-digit): ", n);
    for(int i=0; i<n; i++){
       scanf("%d", &key);
       insert(key);
    }

    display();
    return 0;
}
```

OUTPUT:

```
Enter size of Hash Table (m): 10
Enter number of employee records: 5
Enter 5 employee keys (4-digit):
1234
2345
3456
4567
5678

Hash Table Contents:
HT[0] --> EMPTY
HT[1] --> EMPTY
HT[2] --> EMPTY
HT[3] --> EMPTY
HT[4] --> 1234
HT[5] --> 2345
HT[6] --> 3456
HT[7] --> 4567
HT[8] --> 5678
HT[9] --> EMPTY
```

16/12/25