Rajalakshmi Engineering College

*241901104*

*241901104*

*241901104*

Name: Shafraz.R

*241901104*

Email: 241901104@rajalakshmi.edu.in

Roll no: 241901104

Phone: 6380360049

Branch: REC

Department: l CSE (CS) FB

Batch: 2028

Degree: B.E - CSE (CS)

NeoColab\_REC\_CS23231\_DATA STRUCTURES

*241901104*

*241901104*

REC\_DS using C\_Week 7\_COD\_Question 1

Attempt : 1

Total Mark : 10

Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Ravi is building a basic hash table to manage student roll numbers for quick lookup. He decides to use Linear Probing to handle collisions.

*241901104*

*241901104*

*241901104*

*241901104*

Implement a hash table using linear probing where:

The hash function is: index = roll\_number % table\_sizeOn collision, check subsequent indexes (i+1, i+2, ...) until an empty slot is found.

You need to:

Insert a list of n student roll numbers into the hash table.Print the final state of the hash table. If a slot is empty, print -1.

*Input Format*

*241901104*

*241901104*

*241901104*

*241901104*

The first line of the input contains two integers n and table\_size, where n is the number of roll numbers to be inserted, and table\_size is the size of the hash

*241901104*

The second line contains n space-separated integers — the roll numbers to insert into the hash table.

table.

*241901104*

*241901104*

*241901104*

*Output Format*

The output should print a single line with table\_size space-separated integers representing the final state of the hash table after all insertions.

If any slot remains unoccupied, it should be represented as -1.

*241901104*

*241901104*

Refer to the sample output for formatting specifications.

*241901104*

*241901104*

*Sample Test Case*

Input: 4 7

50 700 76 85

Output: 700 50 85 -1 -1 -1 76

*Answer*

#include <stdio.h>

*241901104*

*241901104*

#define MAX 100

// You are using GCC

void initializeTable(int table[], int size) {

//Type your code here

for (int i =0; i <size; i++) table[i] = -1;

*241901104*

*241901104*

}

int linearProbe(int table[], int size, int num) {

//Type your code here int index = num %size; int start = index; while (table[index]!= -1) { index = (index+ 1)% size; if (index ==start)

*241901104*

*241901104*

*241901104*

*241901104*

return index;

break;

}

}

*241901104*

void insertIntoHashTable(int table[], int size, int arr[], int n) {

*241901104*

*241901104*

*241901104*

//Type your code here

for (int i = 0; i < n; i++) {

int index =linearProbe(table, size, arr[i]); table[index]= arr[i];

}

}

*241901104*

*241901104*

void printTable(int table[], int size) { for (int i =0; i <size;i++) { printf("%d",table[i]); if(i != size -1)

*241901104*

*241901104*

printf(" ");

}

printf("\n");

} int main() { int n, table\_size;

scanf("%d %d", &n, &table\_size);

*241901104*

*241901104*

*241901104*

int arr[MAX];

*241901104*

int table[MAX];

for (int i = 0; i < n; i++)

scanf("%d", &arr[i]);

initializeTable(table, table\_size); insertIntoHashTable(table, table\_size, arr, n); printTable(table, table\_size);

return 0;

*Status :* Correct  *Marks : 10/10*

}

*241901104*

*241901104*

*241901104*

*241901104*