***NAME : JANHAVI GATTANI***

***BATCH: 2***

***PRN : 12311291***

***ROLL NO : 37***

***LAB ASSIGNMENT 15***

**QUESTION:** **Implement following collision handling techniques for Hash table. A. Linear probing. B. Quadratic Probing. C. Double Hashing using Mod as a Hash Function.**

**CODE:**

#include <stdio.h>

#define SIZE 10

#define EMPTY -1

void display(int table[]) {

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

printf("%d ", table[i]);

}

printf("\n");

}

void insertLinear(int table[], int key) {

int index = key % SIZE;

int i = 0;

while (table[(index + i) % SIZE] != EMPTY && i < SIZE) {

i++;

}

if (i < SIZE)

table[(index + i) % SIZE] = key;

else

printf("Linear: Table is full!\n");

}

void insertQuadratic(int table[], int key) {

int index = key % SIZE;

int i = 0;

while (table[(index + i \* i) % SIZE] != EMPTY && i < SIZE) {

i++;

}

if (i < SIZE)

table[(index + i \* i) % SIZE] = key;

else

printf("Quadratic: Table is full!\n");

}

int hash2(int key) {

return 7 - (key % 7);

}

void insertDoubleHash(int table[], int key) {

int index = key % SIZE;

int step = hash2(key);

int i = 0;

while (table[(index + i \* step) % SIZE] != EMPTY && i < SIZE) {

i++;

}

if (i < SIZE)

table[(index + i \* step) % SIZE] = key;

else

printf("Double Hashing: Table is full!\n");

}

int main() {

int linearTable[SIZE], quadraticTable[SIZE], doubleHashTable[SIZE];

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

linearTable[i] = EMPTY;

quadraticTable[i] = EMPTY;

doubleHashTable[i] = EMPTY;

}

int keys[] = {10, 20, 30, 25, 35};

int n = sizeof(keys)/sizeof(keys[0]);

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

insertLinear(linearTable, keys[i]);

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

insertQuadratic(quadraticTable, keys[i]);

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

insertDoubleHash(doubleHashTable, keys[i]);

printf("Linear Probing:\n");

display(linearTable);

printf("\nQuadratic Probing:\n");

display(quadraticTable);

printf("\nDouble Hashing:\n");

display(doubleHashTable);

return 0;

}

**OUTPUT:**

