CAP 770: ADVANCED DATA STRUCTURES

CONTINUOUS ASSESSMENTS (C.A)-4

ST_NAME : - **EKHLAKH AHMAD**
REG NO. : - **12209166**
ROLL NO. : - **RD2215B50**
SECTION : - **D2215**
GROUP : - **2**

Q.1. Write a program to store the numbers in hash table with linear probing.

Ans: -

```
#include <iostream>
using namespace std;
const int TABLE_SIZE = 10;
class HashTable {
private:
   int table[TABLE_SIZE];
public:
   HashTable() {
      for (int i = 0; i < TABLE_SIZE; i++) {
         table[i] = -1;
      }
   }
   void insert(int key) {
      int index = key % TABLE_SIZE;
      int i = 0;
      while (table[index] != -1 && i < TABLE_SIZE) {
         index = (index + 1) % TABLE_SIZE;
         i++;
      }
      if (i == TABLE_SIZE) {
```

```
            cout << "Hash table is full!" << endl;
            return;
        }
        table[index] = key;
    }

    void remove(int key) {
        int index = key % TABLE_SIZE;
        int i = 0;
        while (table[index] != key && i < TABLE_SIZE) {
            index = (index + 1) % TABLE_SIZE;
            i++;
        }
        if (i == TABLE_SIZE) {
            cout << "Key not found!" << endl;
            return;
        }
        table[index] = -1;
    }

    void display() {
        cout << "Hash Table:" << endl;
        for (int i = 0; i < TABLE_SIZE; i++) {
            cout << "[" << i << "]: " << table[i] << endl;
        }
    }
};
int main() {
```

```
    HashTable hashTable;
    hashTable.insert(12);
    hashTable.insert(23);
    hashTable.insert(10);
    hashTable.insert(9);
    hashTable.insert(3);
    hashTable.insert(27);
    hashTable.display();
    hashTable.remove(10);
    hashTable.remove(9);
    hashTable.display();
    hashTable.insert(6);
    hashTable.insert(21);
    hashTable.display();
    return 0;
}
```

OUTPUT