ASSIGNMENT 7

AIM: Insert the keys into a hash table of length m using open addressing using double hashing with h(k)=(1+kmod(m-1)).

OBJECTIVE: To study and learn the concepts of double hashing.

THEORY: Double hashing is a collision resolving technique in **Open Addressed** Hash tables. Double hashing uses the idea of applying a second hash function to key when a collision occurs.

Double hashing can be done using:
(hash1(key) + i * hash2(key)) % TABLE_SIZE
Here hash1() and hash2() are hash functions and TABLE_SIZE
is size of hash table.
(We repeat by increasing i when collision occurs)

First hash function is typically hash1(key) = key % TABLE_SIZE

A popular second hash function is:

hash2(key) = PRIME – (key % PRIME) where PRIME is a prime smaller than the TABLE_SIZE.

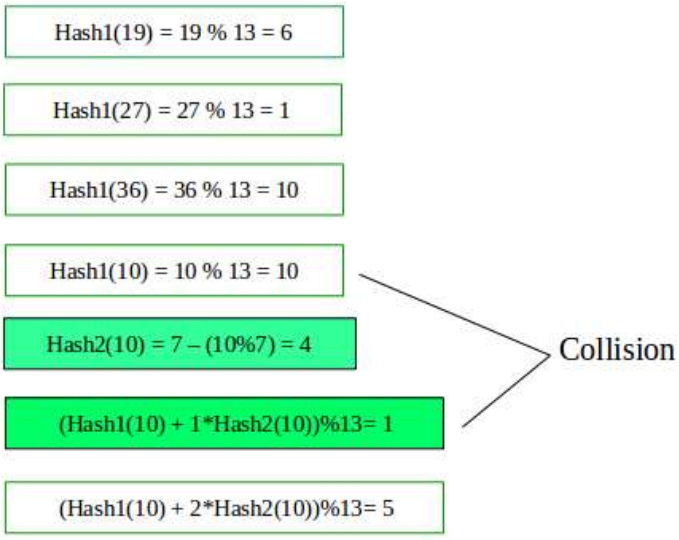
A good second Hash function is:

- It must never evaluate to zero
- Must make sure that all cells can be probed

ALGORITHM:

Lets say, Hash1 (key) = key % 13

Hash2 (key) = 7 – (key % 7)

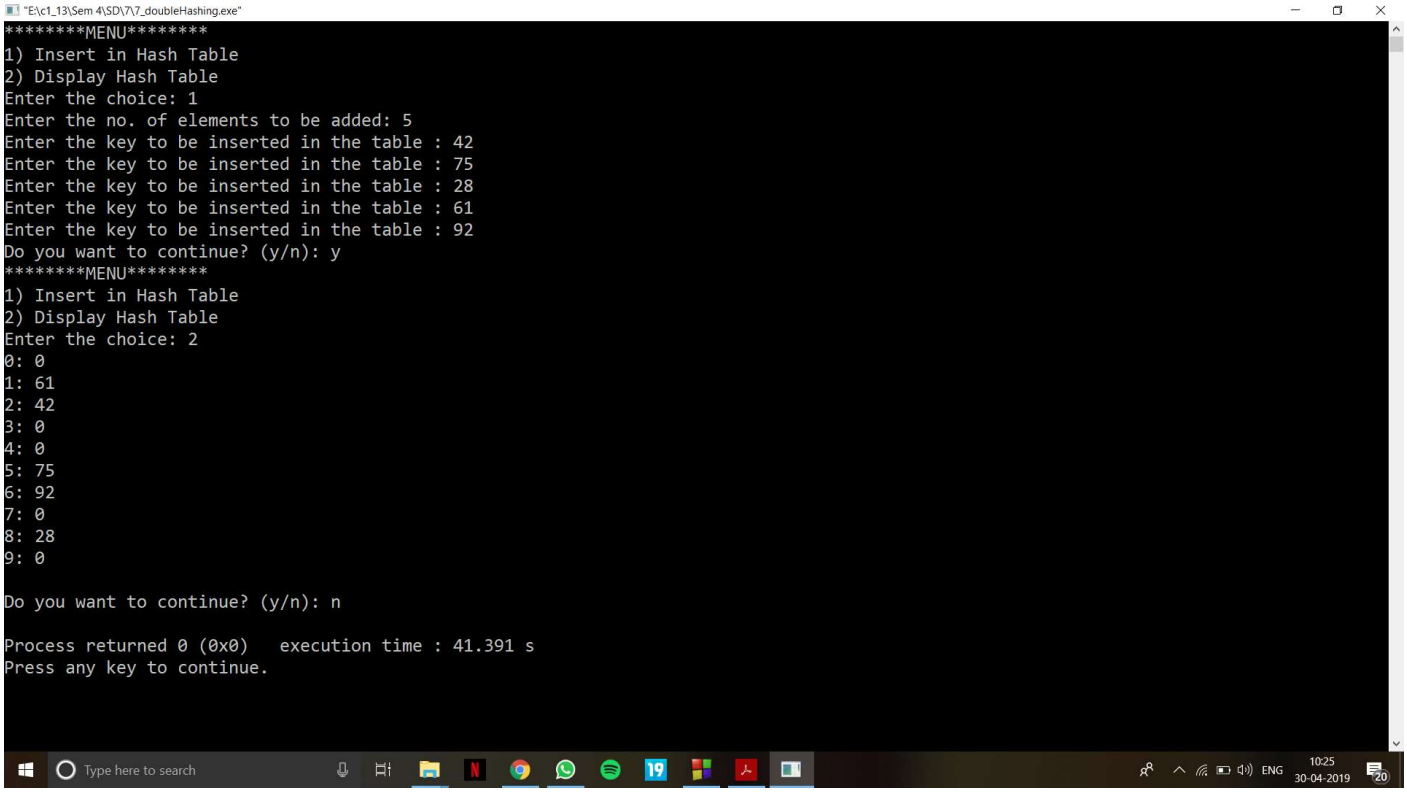


PROGRAM:

```
#include <iostream>

using namespace std;

const int TABLE_SIZE = 10;
int hashTable[TABLE_SIZE] ={0};

void addInTable(){
  int key;
  bool isPlaced = false;
  cout<<"Enter the key to be inserted in the table : ";
  cin>>key;
  int Hash1 = key % TABLE_SIZE;
  int Hash2 = 7 - (key % 7);
  if(hashTable[Hash1] == 0){
    hashTable[Hash1] = key;
    isPlaced = true;
  }
  else if(hashTable[Hash2] == 0){
    hashTable[Hash2] = key;
    isPlaced = true;
  }
  else{
    for(int i = 0;i<TABLE_SIZE; i++){
      if(hashTable[Hash1 + (i*Hash2)] == 0){
        hashTable[Hash1 + (i*Hash2)] = key;
        isPlaced = true;
      }
    }
  }
  if(!isPlaced){
    cout<<"The number is not inserted as array is full."<<endl;
  }
}

void displayTable(){
  for(int i = 0; i<TABLE_SIZE; i++){
    cout<<i<<": "<<hashTable[i]<<endl;
  }
  cout<<endl;
```

```
}

int main()
{
  int choice,n;
  char ch = 'y';
  while(ch=='y' || ch=='Y')
  {
    cout<<"********MENU********"<<endl;
    cout<<"1) Insert in Hash Table"<<endl;
    cout<<"2) Display Hash Table"<<endl;
    cout<<"Enter the choice: ";
    cin>>choice;
    switch(choice){
    case 1:
      cout<<"Enter the no. of elements to be added: ";
      cin>>n;
      while(n != 0)
      {
        addInTable();
        n--;
      }
      break;
    case 2: displayTable();
      break;
    default: cout<<"Wrong choice "<<endl;
    }
    cout<<"Do you want to continue? (y/n): ";
    cin>>ch;
  }
  return 0;
}
```

OUTPUT:



CONCLUSION: We successfully implemented open addressing using double hashing.