Name : Harshal Kodgire
PRN : 2019BTECS00029
Batch : B1
Topic : CNS Assignment 14

Aim : CRT Algorithm

Theory :
Let me write the following set of k equations:
x = a1 (mod n1)
.
.
.
x = ak (mod nk)

This is equivalent to saying that x mod ni = ai (for i=1...k). The notation above is common in group theory, where you can define the group of integers modulo some number n and then you state equivalences (or congruence) within that group. So x is the unknown; instead of knowing x, we know the remainder of the division of x by a group of numbers. If the numbers ni are pairwise coprimes (i.e. each one is coprime with all the others) then the equations have exactly one solution. Such solution will be modulo N, with N equal to the product of all the ni.

Code :

```
#include <bits/stdc++.h>
using namespace std;

// Function for extended Euclidean Algorithm
int ansS, ansT;
int findGcdExtended(int r1, int r2, int s1, int s2, int t1, int t2)
{
    // Base Case
    if (r2 == 0)
    {
        ansS = s1;
        ansT = t1;
        return r1;
    }

    int q = r1 / r2;
    int r = r1 % r2;
```

```
    int s = s1 - q * s2;
    int t = t1 - q * t2;

    cout << q << " " << r1 << " " << r2 << " " << r << " " << s1 << " "
<< s2 << " " << s << " " << t1 << " " << t2 << " " << t << endl;

    return findGcdExtended(r2, r, s2, s, t2, t);
}

int modInverse(int A, int M)
{
    int x, y;
    int g = findGcdExtended(A, M, 1, 0, 0, 1);
    if (g != 1) {
        cout << "\n Inverse doesn't exist";
        return 0;
    }
    else {

        // m is added to handle negative x

        int res = (ansS % M + M) % M;
        cout << "\n Inverse is " << res << endl;
        return res;
    }
}



int findX(vector<int> num, vector<int> rem, int k)
{
    // Compute product of all numbers
    int prod = 1;
    for (int i = 0; i < k; i++)
        prod *= num[i];

    // Initialize result
    int result = 0;

    // Apply above formula
    for (int i = 0; i < k; i++) {
        int pp = prod / num[i];
        result += rem[i] * modInverse(pp, num[i]) * pp;
```

```
    }

    return result % prod;
}



int main()
{

    // 3
    // 3 4 5
    // 2 3 1

    int k;
    cout << "\n Enter total count of equations : ";
    cin >> k;

    vector<int> num(k), rem(k);
    cout<<"\n Enter divisors : ";
    for (int i = 0; i < k; i++)
        cin >> num[i];

    cout<<"\n Enter remainders : ";
    for (int i = 0; i < k; i++)
        cin >> rem[i];

    int x = findX(num, rem, k);
    cout << "\n x is " << x;


    return 0;
}
```

Output :

```
D:\WCE_ENGINEERING\BTECH_SEM1\CNS lab\LA2>g++ assignment13_CRT.cpp

D:\WCE_ENGINEERING\BTECH_SEM1\CNS lab\LA2>a.exe

 Enter total count of equations : 3

 Enter divisors : 3
5
7

 Enter remainders : 1
2
3
11 35 3 2 1 0 1 0 1 -11
1 3 2 1 0 1 -1 1 -11 12
2 2 1 0 1 -1 3 -11 12 -35

 Inverse is 2
4 21 5 1 1 0 1 0 1 -4
5 5 1 0 0 1 -5 1 -4 21

 Inverse is 1
2 15 7 1 1 0 1 0 1 -2
7 7 1 0 0 1 -7 1 -2 15

 Inverse is 1

 x is 52
D:\WCE_ENGINEERING\BTECH_SEM1\CNS lab\LA2>
```