<u>**Cryptography and Network Security**</u>

Name: Piyush Mhaske Batch: B3
PRN : 2019BTECS00089

Assignment 11 : Chinese Reminder Theorem

Problem statement : Chinese Reminder theorem

Theory : ***Chinese Remainder Theorem states that there always exists an x that satisfies given congruences***

```
// This program implements the Chinese Remainder Theorm
//code by :- Piyush Mhaske
#include <bits/stdc++.h>
#define ll long long
#define ul unsigned long long
#define pb emplace_back
#define po pop_back
#define vi vector<ll>
#define vii vector<vector<ll>>
using namespace std;
const int MODVALUE = 1e9;

long long gcdExtended(long long a, long long b, long long *x, long long *y)
{
  cout << a << " " << b << " "
     << " " << *x << " " << *y << "\n";
  // Base Case
  if (b == 0)
  {
    return *x;
  }
  long long q = a / b;
  long long x1 = *y;
  long long y1 = *x - q * (*y);
  long long t1 = gcdExtended(b, a % b, &x1, &y1);

  cout << a << " " << *x << "\n";
  if (*x == 0 && t1 < 0)
```

```
    return a + t1;
  else
    return t1;

  // return gcd;
}


int main() {


    char patternChar = '-';
    char resetChar = ' ';
    int lineWidth = 90;
    int initialWidth = 50;

    cout << setfill(patternChar) << setw(lineWidth) << patternChar << endl;
    cout << setfill(resetChar);
    cout << setw(initialWidth) << "Chinese Remainder Theorm" << endl;
    cout << setfill(patternChar) << setw(lineWidth) << patternChar << endl;
    cout << setfill(resetChar);

    cout << "Enter the total number of equations involved: ";
    int n;
    cin >> n;

    vector<int> divisor(n, 0);
    vector<int> remainder(n, 0);

    // M = m1 * m2 * m3 * .....
    long long int M = 1;

    cout << "Enter the divisors of " << n << " the equations: \n" << endl;
    for(int i = 0; i < n; i++){
        cin >> divisor[i];
        M *= divisor[i];
        M %= MODVALUE;
    }

    cout << "Enter the remainders of " << n << " equations: \n" << endl;
    for(int i = 0; i < n; i++){
        cin >> remainder[i];
    }

    // finding m1, m2, m3, ...
    vector<int> mValues(n);
    vector<int> invMValues(n);

    for(int i = 0; i < n; i++){
        mValues[i] = M/divisor[i];
        long long x=0, y=1;

        x = gcdExtended(divisor[i],mValues[i],&x, &y);
        cout<<"The inverse for M"<<(i+1)<<" = "<<mValues[i]<<" is "<<x<<"\n";
```

```
        invMValues[i] = x;
    }

    long long ans = 0;

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

        ans += (((1LL* remainder[i] * mValues[i])%M)*invMValues[i])%M;
        ans %= M;
    }



    cout << "\n The Value of X is Ans: " << ans << endl;

    return 0;
}
```

Output :

```
PS D:\Academics\Fourth Year\CNS Lab\cns lab> cd "d:\Academics\Fourth Year\CNS Lab\cns lab"
PS D:\Academics\Fourth Year\CNS Lab\cns lab> & .\"assignment11.exe"
-----------------------------------------------------------------------------------------
                          Chinese Remainder Theorm
-----------------------------------------------------------------------------------------
Enter the total number of equations involved: 3
Enter the divisors of 3 the equations:

3 5 7
Enter the remainders of 3 equations:

2 3 2
3 35  0 1
35 3  1 0
3 2  0 1
2 1  1 -1
1 0  -1 3
2 1
3 0
35 1
3 0
The inverse for 3 35 is 2
5 21  0 1
21 5  1 0
5 1  0 1
1 0  1 -5
5 0
21 1
5 0
The inverse for 5 21 is 1
7 15  0 1
15 7  1 0
7 1  0 1
1 0  1 -7
7 0
15 1
5 0
21 1
5 0
The inverse for 5 21 is 1
```

```
The inverse for 5 21 is 1
7 15  0 1
15 7  1 0
7 1  0 1
1 0  1 -7
7 0
15 1
7 0
The inverse for 7 15 is 1

 Ans: 23
PS D:\Academics\Fourth Year\CNS Lab\cns lab> 
```