Government College of Engineering, Jalgaon **(An Autonomous Institute of Govt. of Maharashtra)**	
Name:	**PRN:**
Class: L.Y **Semester:** VII	**Batch:** B
Date of Performance: ______________	**Date of Completion:** _______________
Subject: CO407U CNSL	**Subject Teacher:** Ms. Shrutika Mahajan

Practical – 6

<u>Aim:</u> Implementation of Chinese Remainder Theorem.

<u>Theory:</u>

Introduction:

The Chinese Remainder Theorem (CRT) is a fundamental theorem in number theory used to solve systems of simultaneous congruences (modular equations).
It ensures that if you have several modular equations with pairwise coprime moduli, there exists a unique solution modulo the product of those moduli.

It is very useful in cryptography, especially in algorithms like RSA, for speeding up modular computations.

Working Principle:

If we have a system of congruences:

$$x \equiv a_1 \pmod{m_1}$$
$$x \equiv a_2 \pmod{m_2}$$
$$x \equiv a_3 \pmod{m_3}$$

where m_1, m_2, m_3 are **pairwise coprime**,
then there exists a unique solution x modulo $M = m_1 \times m_2 \times m_3$.

Steps to Solve:

1. Compute $M = m_1 \times m_2 \times m_3$
2. For each i:
$$M_i = M / m_i$$
3. Find the modular inverse y_i of M_i modulo m_i:
$$M_i \times y_i \equiv 1 \pmod{m_i}$$

4. The solution is:

$$x = (a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 y_3) \bmod M$$

Example:
Solve the following system of congruences using the Chinese Remainder Theorem:

$$\boldsymbol{x \equiv 2 \pmod{3}}$$
$$\boldsymbol{x \equiv 3 \pmod{4}}$$
$$\boldsymbol{x \equiv 1 \pmod{5}}$$

Step 1:
Find the product of all moduli:

$$M = 3 \times 4 \times 5 = 60$$

Step 2:
Compute $M_i = \frac{M}{m_i}$ for each modulus:

$$M_1 = 60/3 = 20$$
$$M_2 = 60/4 = 15$$
$$M_3 = 60/5 = 12$$

Step 3:
Find the modular inverse y_i of each M_i modulo m_i:
We need $M_i \times y_i \equiv 1 \pmod{m_i}$

M_i	m_i	Equation	y_i
20	3	$20 \times y_1 \equiv 1 \pmod{3} \rightarrow 2 \times y_1 \equiv 1$	$y_1 = 2$
15	4	$15 \times y_2 \equiv 1 \pmod{4} \rightarrow 3 \times y_2 \equiv 1$	$y_2 = 3$
12	5	$12 \times y_3 \equiv 1 \pmod{5} \rightarrow 2 \times y_3 \equiv 1$	$y_3 = 3$

Step 4:
Compute:

$$x = (a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 y_3) \bmod M$$

Substitute values:

$$x = (2 \times 20 \times 2 + 3 \times 15 \times 3 + 1 \times 12 \times 3) \bmod 60$$
$$x = (80 + 135 + 36) \bmod 60$$
$$x = 251 \bmod 60 = 11$$
$$\boxed{x = 11}$$

Hence, $x \equiv 11 \pmod{60}$.

Code:

```
package string;
import java.util.*;
public class CRT {
    static int findInverse(int a, int m) {
        a = a % m;
        for (int x = 1; x < m; x++)
            if ((a * x) % m == 1)
                return x;
        return 1;
    }
    static int findX(int[] num, int[] rem, int k) {
        int prod = 1;
        for (int i = 0; i < k; i++)
            prod *= num[i];

        int result = 0;
        for (int i = 0; i < k; i++) {
            int pp = prod / num[i];
            result += rem[i] * findInverse(pp, num[i]) * pp;
        }
        return result % prod;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of equations: ");
        int n = sc.nextInt();

        int[] num = new int[n];  // moduli
        int[] rem = new int[n];  // remainders

        System.out.println("\nEnter the moduli (must be pairwise coprime):");
        for (int i = 0; i < n; i++) {
            System.out.print("m" + (i + 1) + ": ");
            num[i] = sc.nextInt();
        }
        System.out.println("\nEnter the remainders:");
        for (int i = 0; i < n; i++) {
```

```
            System.out.print("a" + (i + 1) + ": ");
            rem[i] = sc.nextInt();
        }

        int x = findX(num, rem, n);

        int prod = 1;
        for (int val : num) prod *= val;

        System.out.println("\nThe value of x is: " + x);
        System.out.println("x ≡ " + x + " (mod " + prod + ")");
        sc.close();
    }
}
```

Output:

```
Enter number of equations: 3

Enter the moduli (must be pairwise coprime):
m1: 3
m2: 4
m3: 5

Enter the remainders:
a1: 2
a2: 3
a3: 1

The value of x is: 11
x ≡ 11 (mod 60)
```

Applications:

1. Used in RSA algorithm to speed up encryption and decryption.
2. Error correction and coding theory.
3. Efficient computation in modular arithmetic systems.

Conclusion:

- The **Chinese Remainder Theorem** provides an efficient method for solving modular arithmetic problems.

- It guarantees a **unique solution** when the moduli are **pairwise coprime**.
- It is widely used in **RSA decryption** to improve performance by working with smaller moduli.

Course Teacher
Ms. Shrutika Mahajan