**Final Year B.Tech. (CSE) – VII [ 2024-25]**

**6CS451: Cryptography and Network Security Lab (C&NS Lab)**

**Date: 19/08/2024**

**Assignment 4**

**PRN:** 21510042  **Name:** Omkar Rajesh Auti

1. **Implementation of Chinese Remainder Theorem (CRT)**

**Ans:**

The Chinese Remainder Theorem (CRT) is a powerful tool in number theory that provides a solution to a system of simultaneous congruences with pairwise coprime moduli. Given a system of congruences, the CRT allows us to find a unique solution modulo the product of the moduli.

**Problem Description**

Given n congruences: x≡a1 (mod m1), x≡a2 (mod m2) ⋮x ≡ an (mod mn)

Where the moduli m1, m2, …, mn are pairwise coprime, the CRT provides a unique solution modulo M=m1×m2×⋯× mn.

For each congruence x ≡ ai (mod mi), it calculates the partial solution using the formula: x ≡ ai × Mi × inverse(Mi, mi) (mod M) where Mi=M/mi

The final solution is obtained by summing all partial solutions modulo M.

**Python code:**

def extended\_euclidean\_algorithm(a, b):

    """

    Compute the GCD of a and b, as well as the coefficients x and y

    such that ax + by = gcd(a, b) using the Extended Euclidean algorithm.

    Parameters:

    a (int): First integer.

    b (int): Second integer.

    Returns:

    tuple: (gcd, x, y) where gcd is the GCD of a and b, and x, y are

    the coefficients of Bézout's identity.

    """

    if b == 0:

        return a, 1, 0

    else:

        gcd, x1, y1 = extended\_euclidean\_algorithm(b, a % b)

        x = y1

        y = x1 - (a // b) \* y1

        return gcd, x, y

def chinese\_remainder\_theorem(a, m):

    """

    Solve the system of congruences using the Chinese Remainder Theorem.

    Parameters:

    a (list): List of remainders.

    m (list): List of moduli (must be pairwise coprime).

    Returns:

    int: The smallest non-negative solution to the system of congruences.

    """

    assert len(a) == len(m), "The number of remainders and moduli must be the same"

    # Calculate the product of all moduli

    M = 1

    for mi in m:

        M \*= mi

    # Initialize the solution

    x = 0

    # Apply the CRT

    for ai, mi in zip(a, m):

        Mi = M // mi  # M\_i = M / m\_i

        gcd, inverse, \_ = extended\_euclidean\_algorithm(Mi, mi)

        if gcd != 1:

            raise ValueError("Moduli are not pairwise coprime")

        x += ai \* inverse \* Mi

    return x % M

def main():

    """

    The main function to run the program.

    """

    while True:

        print("\nChinese Remainder Theorem (CRT)")

        print("1. Solve System of Congruences")

        print("2. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':

            n = int(input("\nEnter the number of congruences: "))

            a = []

            m = []

            for i in range(n):

                ai = int(input(f"\nEnter remainder a[{i+1}]: "))

                mi = int(input(f"Enter modulus m[{i+1}]: "))

                a.append(ai)

                m.append(mi)

            solution = chinese\_remainder\_theorem(a, m)

            print(f"\nThe solution to the system of congruences is: {solution}")

        elif choice == '2':

            print("Exiting the program.")

            break

        else:

            print("Invalid choice. Please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Output:**

