**Cryptography & Network Security Lab**

**PRN/ Roll No: 2019BTECS00090**

**Full name: Udaykumar Gadikar**

**Assignment No. 10**

**Title: Chinese Remainder Theorem**

**Aim: To Demonstrate Chinese Remainder Theorem**

**Theory:**

**In mathematics, the Chinese remainder theorem states that if one knows the remainders of the Euclidean division of an integer n by several integers, then one can determine uniquely the remainder of the division of n by the product of these integers, under the condition that the divisors are pair wise co-prime.**

**Code:**

def Mod\_Inv(a, b):

    t1 = 0

    t2 = 1

    c = b

    d = a

    while (b != 0):

        q = a // b

        r = a % b

        a = b

        b = r

        t = t1 - (q \* t2)

        t1 = t2

        t2 = t

    if (t1 < 0):

        t1 = t1 + d

    return t1

def findMinX(num, rem, k):

    prod = 1

    for i in range(0, k):

        prod = prod \* num[i]

    print(prod)

    result = 0

    for i in range(0, k):

        pp = prod // num[i]

        result = result + rem[i] \* Mod\_Inv(pp, num[i]) \* pp

    return result % prod

# num = [25, 4]

# rem = [129934811447123020117172145698449, 129934811447123020117172145698449]

# x = 129934811447123020117172145698449(mod 25)

# x = 129934811447123020117172145698449(mod 4)

n = int(input("Enter n: "))

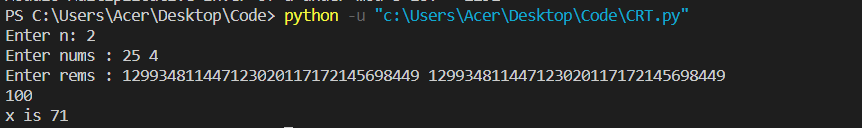
rem = []

num = list(map(int, input("Enter nums : ").strip().split()))[:n]

rem = list(map(int, input("Enter rems : ").strip().split()))[:n]

print("x is", findMinX(num, rem, n))

**Output:**

****

**Conclusion:**

**The Chinese remainder theorem is widely used for computing with large integers, as it allows replacing a computation for which one knows a bound on the size of the result by several similar computations on small integers.**