import java.util.Scanner;

public class RSa {

    public static void main(String[] args) {

        String pl, cr, dr;

        int x, y, p, q, n, m, e, d;

        try (Scanner sc = new Scanner(System.in)) {

            System.out.println("Enter the upper limit for p:");

            x = sc.nextInt();

            System.out.println("Enter the upper limit for q:");

            y = sc.nextInt();

            p = findLargestPrime(x);

            System.out.println("The largest prime number (p) is: " + p);

            q = findLargestPrime(y);

            System.out.println("The largest prime number (q) is: " + q);

            n = p \* q;

            m = (p - 1) \* (q - 1);

            System.err.println("The value of (n) is:" + n);

            System.err.println("The value of (m) is:" + m);

            e = findCoprime(m);

            System.err.println("The value of (e) is:" + e);

            d = modInverse(e, m);

            System.err.println("The value of (d) is:" + d);

            System.err.println("Enter the plain text:");

            pl = sc.next();

        }

        System.err.println("The plain text is:" + pl);

        cr = encrypt(pl, e, n);

        System.err.println("The cipher text is:" + cr);

        dr = decrypt(cr, d, n);

        System.err.println("The decrypted text is:" + dr);

    }

    public static int findLargestPrime(int limit) {

        for (int i = limit; i >= 2; i--) {

            if (isPrime(i)) {

                return i;

            }

        }

        return 2;

    }

    public static boolean isPrime(int num) {

        if (num <= 1) {

            return false;

        }

        for (int i = 2; i <= Math.sqrt(num); i++) {

            if (num % i == 0) {

                return false;

            }

        }

        return true;

    }

    public static int gcd(int a, int b) {

        while (b != 0) {

            int temp = b;

            b = a % b;

            a = temp;

        }

        return a;

    }

    public static int findCoprime(int m) {

        for (int e = 2; e < m; e++) {

            if (gcd(e, m) == 1) {

                return e;

            }

        }

        throw new ArithmeticException("No coprime found");

    }

    public static int modInverse(int a, int m) {

        for (int x = 1; x < m; x++) {

            if ((a \* x) % m == 1) {

                return x;

            }

        }

        throw new ArithmeticException("Modular inverse does not exist");

    }

    public static String encrypt(String pl, int e, int n) {

        int i, j, k;

        String cr = "";

        char c;

        for (i = 0; i < pl.length(); i++) {

            c = pl.charAt(i);

            j = (int) c;

            k = 1;

            for (int l = 0; l < e; l++) {

                k = k \* j;

                k = k % n;

            }

            cr = cr + (char) k;

        }

        return cr;

    }

    public static String decrypt(String cr, int d, int n) {

        int i, j, k;

        String dr = "";

        char c;

        for (i = 0; i < cr.length(); i++) {

            c = cr.charAt(i);

            j = (int) c;

            k = 1;

            for (int l = 0; l < d; l++) {

                k = k \* j;

                k = k % n;

            }

            dr = dr + (char) k;

        }

        return dr;

    }

}