**CSS PRACTICAL 2**

import math

def is\_coprime(a, b):

    return math.gcd(a, b) == 1

def mod\_inverse(a, m):

    for x in range(1, m):

        if (a \* x) % m == 1:

            return x

    return None

def generate\_keypair(p, q):

    n = p \* q

    phi\_n = (p - 1) \* (q - 1)

    e = 2

    while e < phi\_n:

        if is\_coprime(e, phi\_n):

            break

        e += 1

    d = mod\_inverse(e, phi\_n)

    return ((n, e), (n, d))

def encrypt(public\_key, plaintext):

    n, e = public\_key

    return (plaintext \*\* e) % n

def decrypt(private\_key, ciphertext):

    n, d = private\_key

    return (ciphertext \*\* d) % n

def main():

    p = int(input("Enter the first prime number (p): "))

    q = int(input("Enter the second prime number (q): "))

    public\_key, private\_key = generate\_keypair(p, q)

    print("Public key (n, e):", public\_key)

    print("Private key (n, d):", private\_key)

    choice = input("Enter 'E' to encrypt or 'D' to decrypt: ").upper()

    if choice == 'E':

        plaintext = int(input("Enter the plaintext to encrypt: "))

        ciphertext = encrypt(public\_key, plaintext)

        print("The encrypted ciphertext is:", ciphertext)

    elif choice == 'D':

        ciphertext = int(input("Enter the ciphertext to decrypt: "))

        decrypted\_text = decrypt(private\_key, ciphertext)

        print("The decrypted plaintext is:", decrypted\_text)

    else:

        print("Invalid choice. Please enter 'E' or 'D.")

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

    main()







