ICS 4 RSA

41323 Gayatri

In [1]:

```
# def gcd(a, b):

#   if a == 0:
#     return b
#   return gcd(b % a, a)

import math

def gcd(a, b):
    m=math.gcd(a,b)
    return m


def mod_pow(a,b,m):
    if b==0:
        return 1
    r=mod_pow(a,b//2,m)
    r=(r*r)%m
    if b%2==1:
        r = (r * a) % m
    return r



P = 53
Q = 59

n = P * Q
phi_n = (P-1) * (Q-1)


e = 2
while e < phi_n:


    if gcd(e, phi_n) == 1:


        break
    e += 1

print("E",e)


k = 1
while (k * phi_n + 1) % e != 0:


    k += 1
d = (k * phi_n + 1) // e

U = [e, n] # Public key
R = [d, n] # Private key



print("Primes:\t\t", P, ",", Q)
print("N:\t\t", n)
print("phi(N):\t\t", phi_n)
```

```
print("e:\t\t", e)
print("d:\t\t", d)
print("Public key:\t", "[e, n] =", U)
print("Private key:\t", "[d, n] =", R)



def encrypt(P, U):
    e, n = U
    c = mod_pow(P, e, n)
    return c

def decrypt(C, R):


    d, n = R
    ret = mod_pow(C, d, n)
    return ret

plaintext = 89

c = encrypt(plaintext, U)
p = decrypt(c, R)
assert(p == plaintext)
```

```
E 3
Primes:          53 , 59
N:               3127
phi(N):          3016
e:               3
d:               2011
Public key:      [e, n] = [3, 3127]
Private key:     [d, n] = [2011, 3127]
```

In [ ]:

```

```

In [ ]:

```

```