Ishaan Mehta E18CSE069 EB02 LabWeek5

In [3]:
```python
import sympy
import random
class KeyGen:

    '''
        This class has methods to generate both the public and private key for asymmetric
        encryption. The standard used for the calculation is as follows:
                Generate two large random primes, p and q
                Compute n=pxq
                Compute ϕ=(p-1)(q-1)
                Choose an integer e, 1<e<ϕ, such that gcd(e,ϕ)=1
                Compute the secret exponent d, 1<d<ϕ, such that ed≡1modϕ
                The public key= (e, n)
                The private key (d, n)
    '''

    e = None

    def __init__(self, bits):
        self.bits = bits
        self.p, self.q = sympy.randprime(
            2**(bits-1), 2**bits), sympy.randprime(2**(bits-1), 2**bits)
        self.n = self.p*self.q
        self.fi = (self.p-1)*(self.q-1)
        self.puK = self.getPublicKey()
        self.prK = self.getPrivateKey()

    def getKeys(self):
        return self.puK, self.prK

    def getPublicKey(self):
        # public key is (e, n)

        # computing all e's s.t gcd(e,fi)=1
        # taking out a random e
        while True:
            randNum = random.randrange(2**(self.bits-1), 2**self.bits)
            if self.gcd(randNum, self.fi) == 1:
                self.e = randNum
                break

        # returning the public key
        return (self.e, self.n)

    def getPrivateKey(self):
        # private key is (d, n)
        assert self.e != None, 'Generate Public Key First'

        # calculating 'd'
        d = self.findModInverse(self.e, self.fi)
        # returning private key
        return (d, self.n)

    def gcd(self, a, b):
        while a != 0:
            a, b = b % a, a
        return b

    def findModInverse(self, a, m):
        if self.gcd(a, m) != 1:
            return None
        u1, u2, u3 = 1, 0, a
        v1, v2, v3 = 0, 1, m

        while v3 != 0:
            q = u3 // v3
            v1, v2, v3, u1, u2, u3 = (
                u1 - q * v1), (u2 - q * v2), (u3 - q * v3), v1, v2, v3

        return u1 % m


if __name__ == '__main__':
    Generator = KeyGen(1024)
    publicKey, privateKey = Generator.getKeys()
    print("----OUTPUT----")
    print(f'''
    Public Key: {publicKey}
    Length of Public Key: {len(str(publicKey[0]))}
    Private Key: {privateKey}
    Length of Private Key: {len(str(privateKey[0]))}
    ''')
```

```
----OUTPUT----

    Public Key: (1111430642308066891483283189267586514476561317567530863725992518648842814053393460745
987253023901349730276380586237165294938152978100937632204718833510139767758673911599896528101885604776
943666062814567968739022013646299689664829118006363590551356558971268656437984686018802761149701096230
69650746508100011913, 150472188571403322315516981915409571289857206709690508519133731866911651303477008
758238049496906451491062954395435635691571102708723135321818043255126684734370429647232753427817443562
008230670273094150348013356999054093332174441967844137075684174057968989506596652855839432942250746071
051029664319178583263076413490515532669899058466932014681530683073269897419367507738959711423695904831
263324265129833776815365955706878199857393864497840126412952923817154775695246877724760332605591734865
764805631958517796173436348272709784438825469085263178305104966694159048099963701747079369873060473164
6440283942865084046569830 7)
    Length of Public Key: 309
    Private Key: (7727955046420883566708231131728902531067966668361309286574735019520432166112228150562
645243992496853625961133046287496235367015221605811999735408295596297430684394653702034580425477583384
820523966529468617046870070997279866820478757923156531038291232678484482343635481999727230320175485996
628051990262493099459221602263848716941378975968050824804366936176017257096218152039306119033688707859
883451783581786545251849278570710472085189564529033525629483088764059557831085333151003792582428874877
606182126676198085171241918694079054456161583989362013001777335111418715223872790097683134444764116944
559117300734779853097, 15047218857140332231551698191540957128985720670969050851913373186691165130347700
875823804949690645149106295439543563569157110270872313532181804325512668473437042964723275342781744356
200823067027309415034801335699905409333217444196784413707568417405796898950659665285583943294225074607
105102966431917858326307641349051553266989905846693201468153068307326989741936750773895971142369590483
126332426512983377681536595570687819985739386449784012641295292381715477569524687772476033260559173486
576480563195851779617343634827270978443882546908526317830510496669415904809996370174707936987306047316
46440283942865084046569830 7)
    Length of Private Key: 616
```

In [ ]: