**Assignment - 10**

**Aim :** To Implement ElGamal cryptosystem.

**Theory:**

In cryptography, the ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie–Hellman key exchange. It was described by Taher Elgamal in 1985. ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems.

**Program:**

import random

from math import pow

a = random.randint(2, 10)

def gcd(a, b):

if a < b:

return gcd(b, a)

elif a % b == 0:

return b;

else:

return gcd(b, a % b)

# Generating large random numbers

def gen\_key(q):

key = random.randint(pow(10, 20), q)

while gcd(q, key) != 1:

key = random.randint(pow(10, 20), q)

return key

# Modular exponentiation

def power(a, b, c):

x = 1

y = a

while b > 0:

if b % 2 == 0:

x = (x \* y) % c;

y = (y \* y) % c

b = int(b / 2)

return x % c

# Asymmetric encryption

def encrypt(msg, q, h, g):

en\_msg = []

k = gen\_key(q)# Private key for sender

s = power(h, k, q)

p = power(g, k, q)

for i in range(0, len(msg)):

en\_msg.append(msg[i])

print("g^k used : ", p)

print("g^ak used : ", s)

for i in range(0, len(en\_msg)):

en\_msg[i] = s \* ord(en\_msg[i])

return en\_msg, p

def decrypt(en\_msg, p, key, q):

dr\_msg = []

h = power(p, key, q)

for i in range(0, len(en\_msg)):

dr\_msg.append(chr(int(en\_msg[i]/h)))

return dr\_msg

# Driver code

def main():

msg = input ("Enter the message to be encrypted: ");

print("Original Message :", msg)

q = random.randint(pow(10, 20), pow(10, 50))

g = random.randint(2, q)

key = gen\_key(q)# Private key for receiver

h = power(g, key, q)

print("g used : ", g)

print("g^a used : ", h)

en\_msg, p = encrypt(msg, q, h, g)

dr\_msg = decrypt(en\_msg, p, key, q)

dmsg = ''.join(dr\_msg)

print("Decrypted Message :", dmsg);

# call the main function

main()

**Output:**

