**EXPERIMENT NO. 4**

**Code:**

import math

print("RSA ENCRYPTOR/DECRYPTOR")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("PLEASE ENTER THE 'p' AND 'q' VALUES BELOW:")

p = int(input("Enter a prime number for p: "))

q = int(input("Enter a prime number for q: "))

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

def prime\_check(a):

if a == 2:

return True

elif a < 2 or a % 2 == 0:

return False

for i in range(3, int(a\*\*0.5) + 1, 2): # Check divisibility up to sqrt(a)

if a % i == 0:

return False

return True

check\_p = prime\_check(p)

check\_q = prime\_check(q)

while not (check\_p and check\_q):

print("Both p and q should be prime numbers.")

p = int(input("Enter a prime number for p: "))

q = int(input("Enter a prime number for q: "))

check\_p = prime\_check(p)

check\_q = prime\_check(q)

# RSA Modulus

n = p \* q

print("RSA Modulus(n) is:", n)

# Euler's Totient

r = (p - 1) \* (q - 1)

print("Euler's Totient(r) is:", r)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

def eea(a, b):

if a % b == 0:

return b, 0, 1

else:

gcd, s, t = eea(b, a % b)

s = s - (a // b) \* t

return gcd, t, s

def mult\_inv(e, r):

gcd, s, \_ = eea(e, r)

if gcd != 1:

return None

if s < 0:

s = s % r

return s

# e Value Calculation (Choosing e)

e = 7

print("The value of e is:", e)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

d = mult\_inv(e, r)

if d is None:

print("No multiplicative inverse found. Try different values for p and q.")

exit()

public = (e, n)

private = (d, n)

print("Private Key is:", private)

print("Public Key is:", public)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

def encrypt(pub\_key, n\_text):

e, n = pub\_key

x = []

for i in n\_text:

m = i

c = (m \*\* e) % n

x.append(c)

return x

def decrypt(priv\_key, c\_text):

d, n = priv\_key

x = ''

for i in c\_text:

m = (i \*\* d) % n

x += str(m) + ',' # Join encrypted numbers with commas for display

return x[:-1] # Remove trailing comma

message = input("What would you like encrypted or decrypted? (Enter numbers separated by commas): ")

message = list(map(int, message.split(','))) # Convert the input string into a list of integers

print("Your message is:", message)

choose = input("Type '1' for encryption and '2' for decryption: ")

if choose == '1':

enc\_msg = encrypt(public, message)

print("Your encrypted message is:", enc\_msg)

print("Thank you for using the RSA Encryptor. Goodbye!")

elif choose == '2':

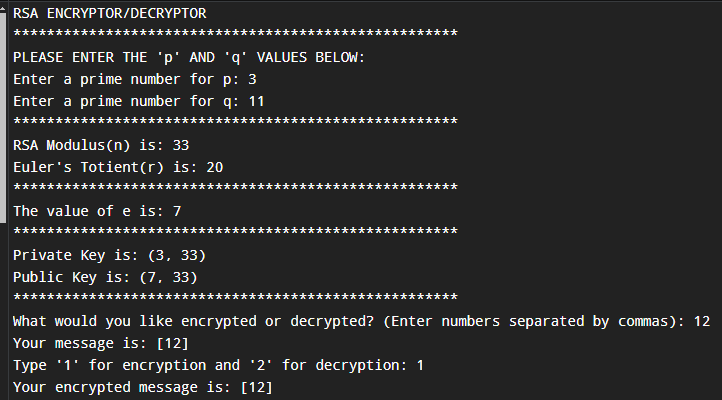
print("Your decrypted message is:", decrypt(private, list(map(int, message.split(',')))))

print("Thank you for using the RSA Encryptor. Goodbye!")

else:

print("You entered the wrong option.")

**Output:**

****