Information Security & Cryptography

Nehal Jhajharia
Lab Assignment 5

Using RSA, construct a program to encrypt and decrypt plaintext messages strings.

```
def gcd(a, h):
   temp = 0
   while(1):
       temp = a % h
       if (temp == 0):
           return h
       a = h
       h = temp

def gcdExtended(a, b):
   # Base Case
   if a == 0 :
       return b,0,1

   gcd,x1,y1 = gcdExtended(b%a, a)

   # Update x and y using results of recursive call
   x = y1 - (b//a) * x1
   y = x1

   return gcd,x,y


# p = 10333
# q = 11621
p =
32317006071311007300714876688669951960444102669715484032130345427524655138867890893197
20141152291346368871796092189801949411955915049092109508815238644828312063087736730099
60917501977503896521067960576383840675682767922186426197561618380943384761704705816458
52036305042887575891541065808607552399123930385521914333389668342420684974786564569494
85617603532632205807780565933102619270846031415025859286417711672594360371846185735759
```

```
835115230164590440369761323328723122712568471082020972515710172693132346967854258065669
79350459972683529986382155251663896479601269392498066254407006858194695899383843569518
33568218188663
q =
323170060713110073007148766886699519604441026697154840321303454275246551388678908931972
014115229134636887179609218980194941195591504909210950881523864482831206308773673009960
917501977503896521067960576383840675682767922186426197561618380943384761704705816458520
363050428875758915410658086075523991239303855219143333896683424206849747865645694948561
760353263220580778056593310261927084603141502585928641771167259436037184618573575983511
523340639947855803707216654176622128812031049459145511400081473963578867676698200428287
937085882522470310920711555402247510310642532098840992381846882464674894987213364501338
89385773

if p == q:
   print('equal')
else:
   print('unequal')

n = p*q

phi = (p-1)*(q-1)
e = 2
while (e < phi):
   if(gcd(e, phi) == 1):
       break
   else:
       e = e+1

# print(e)
# Private key (d stands for decrypt), choosing d such that it satisfies
# d*e = 1 mod(totient)

# k = 2
# d = (1 + (k*phi)) / e
_, _, d = gcdExtended(phi, e)
if d < 0: d += phi
# print(d)

# print((d * e) % phi)
# Encryption
```

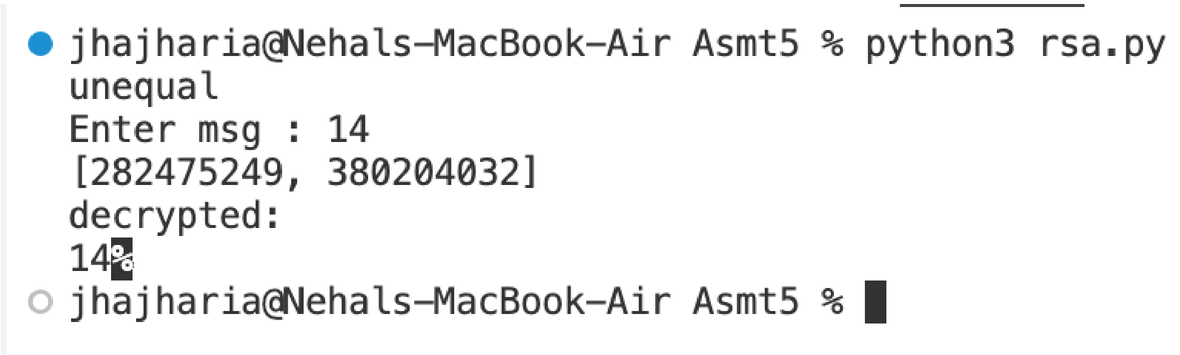
```
msg = (input("Enter msg : "))

res = []
for ch in msg:
   c = ord(ch)
   c = pow(c, e, n)
   res.append(c)

# Encryption c = (msg ^ e) % n
print(res)

# Decryption m = (c ^ d) % n
print("decrypted: ")
for ec in res:
   m = pow(ec, d, n)
   print(chr(m), end="")

# print("Decrypted message = ", m)
```



```
● jhajharia@Nehals-MacBook-Air Asmt5 % python3 rsa.py
  unequal
  Enter msg : 14
  [282475249, 380204032]
  decrypted:
  14%
○ jhajharia@Nehals-MacBook-Air Asmt5 %
```