Name: Jash Mavani

Roll No: 19BCE123

Batch: EL2

Course Name & Course Code: 2CSDE93 & Blockchain Technology

Practical-1

Aim: - To implement digital signature to sign and verify authenticated user. Also, show a message when tampering is detected.

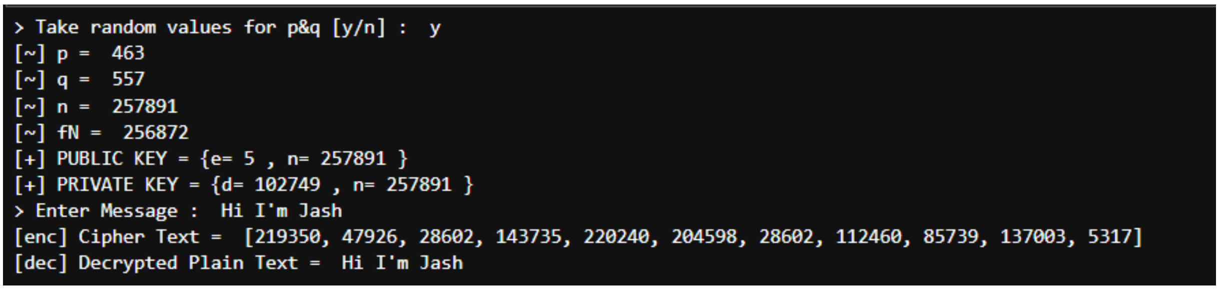
Code:-

```
import random


def check_prime(num):
  if num > 1:
    for i in range(2,num):
        if (num % i) == 0:
            return False
            break
    else:
        return True
  else:
    return False


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

```
random_primes=[947,349,11,463,397,103,401,431,823,881,827,643,197,521,541,769,313,977,557,653,647,43,61,241,419,109,911,601,193,701,967,337,719,467,821,317,73,859,479,353,421,631,113,79,433,23,97,41,167,641,17,137,941,983,661,919,733,181,709,691,659,67,71,383,619,163,839,31,409,587,439,853,461,607,797,239,739,331,449,509,7,271,727,251,29,773,199,613,599,443,491,367,593,563,683,277,929,191,757]
option=input("> Take random values for p&q [y/n] : ")
if option=="n":
  while True:
    p=int(input("> Enter p : "))
    q=int(input("> Enter q : "))
    if(check_prime(p)== True and check_prime(q)==True):
      break
    else:
      print("[ERROR] : p and q must be prime numbers.")
else:
  p=random.choice(random_primes)
  print("[~] p = ",p)
  q=random.choice(random_primes)
  print("[~] q = ",q)

n=p*q
fN=(p-1)*(q-1)
for i in range(2,fN+1):
  if(gcd(i,fN)==1):
    e=i
    break
print("[~] n = ",n)
print("[~] fN = ",fN)
print("[+] PUBLIC KEY = {e=",e,", n=",n,"}");
for i in range(2,fN+1):
  if(((i*e)%fN)==1):
    d=i
    break
print("[+] PRIVATE KEY = {d=",d,", n=",n,"}");12
message=input("> Enter Message : ")
ct=[]
pt=[]
#number encryption
if message.strip().isdigit():
  while int(message)>n:
    print("[ERROR] value of message must be lesser than n 
got",message)
    message=input("> Enter Message : ")

  ct = pow(int(message),e)%n
  print("[enc] Cipher Text = ",ct)
  pt = pow(ct,d)%n
  print("[dec] Decrypted Plain Text = ",pt)
#string encryption
else:
  e_flag=0
  if len(message)>0:
    #encryption
    for i in range(len(message)):
      if ord(message[i])<n:
        ct.append(pow(ord(message[i]),e)%n)
      else:
        print("[ERROR] value of message must be lesser than n 
got",ord(message[i]))
        e_flag=1
        break
    if e_flag==0:
```

```
print("[enc] Cipher Text = ",ct)

#decryption

for i in range(len(message)):

  pt.append(chr(pow(ct[i],d)%n))

print("[dec] Decrypted Plain Text = ",''.join(pt))
```

Output :-



```
> Take random values for p&q [y/n] :  y
[~] p =  463
[~] q =  557
[~] n =  257891
[~] fN =  256872
[+] PUBLIC KEY = {e= 5 , n= 257891 }
[+] PRIVATE KEY = {d= 102749 , n= 257891 }
> Enter Message :  Hi I'm Jash
[enc] Cipher Text =  [219350, 47926, 28602, 143735, 220240, 204598, 28602, 112460, 85739, 137003, 5317]
[dec] Decrypted Plain Text =  Hi I'm Jash
```