# Assignment 8
# U20CS135

Implement the Signature scheme- Digital Signature Standard using RSA.

CODE:

```
# Function to find gcd
# of two numbers
def euclid(m, n):

    if n == 0:
        return m
    else:
        r = m % n
        return euclid(n, r)


# Program to find
# Multiplicative inverse
def exteuclid(a, b):
```

```
r1 = a
r2 = b
s1 = int(1)
s2 = int(0)
t1 = int(0)
t2 = int(1)

while r2 > 0:

    q = r1//r2
    r = r1-q * r2
    r1 = r2
    r2 = r
    s = s1-q * s2
    s1 = s2
    s2 = s
    t = t1-q * t2
    t1 = t2
    t2 = t

if t1 < 0:
```

```
            t1 = t1 % a

    return (r1, t1)

# Enter two large prime
# numbers p and q
p = 823
q = 953
n = p * q
Pn = (p-1)*(q-1)

# Generate encryption key
# in range 1<e<Pn
key = []

for i in range(2, Pn):

    gcd = euclid(Pn, i)

    if gcd == 1:
        key.append(i)
```

```
# Select an encryption key
# from the above list
e = int(313)

# Obtain inverse of
# encryption key in Z_Pn
r, d = exteuclid(Pn, e)
if r == 1:
      d = int(d)
      print("decryption key is: ", d)

else:
      print("Multiplicative inverse for\
      the given encryption key does not \
      exist. Choose a different encryption key ")



# Enter the message to be sent
M = 19070

# Signature is created by Alice
```

```
S = (M**d) % n

# Alice sends M and S both to Bob
# Bob generates message M1 using the
# signature S, Alice's public key e
# and product n.
M1 = (S**e) % n

# If M = M1 only then Bob accepts
# the message sent by Alice.

if M == M1:
    print("As M = M1, Accept the\
    message sent by Alice")
else:
    print("As M not equal to M1,\
    Do not accept the message\
    sent by Alice ")
```