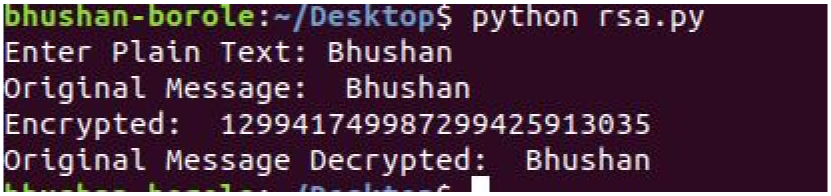
CODE:

```
import math

def egcd(a, b):
    x,y, u,v = 0,1, 1,0
    while a != 0:
        q, r = b//a, b%a
        m, n = x-u*q, y-v*q
        b,a, x,y, u,v = a,r, u,v, m,n
    gcd = b
    return gcd, x, y

def modinv(a, m):
    gcd, x, y = egcd(a, m)
    if gcd != 1:
        return None  # modular inverse does not exist
    else:
        return x % m

def generate_private_key(p, q):
    phi = (p - 1) * (q - 1)
    e = generate_e(phi)
    d = modinv(e, phi)
    return round(d), e

def generate_e(phi, e=2):
    while e < phi:
        if math.gcd(e, phi) == 1:
            break
        else:
            e += 1
    return e

def encrypt(m, e, n):
    return m**e % n



def decrypt(c, d, n):
    return c**d % n
```

```
def rsa(text):
    p = 59
    q = 61
    n = p * q
    d, e = generate_private_key(p, q)

    numbers_of_letters = []
    for t in text:
        numbers_of_letters.append(ord(t) - ord('A'))

    print('Original Message: ', text)
    encrypted = [encrypt(m,e,n) for m in numbers_of_letters]

    print('Encrypted: ', ''.join([str(x) for x in encrypted]))

    decrypted = [decrypt(c,d,n) for c in encrypted]
    print('Original Message Decrypted: ', ''.join([chr(x+65) for x in decrypted]))


if __name__ == '__main__':
    plain_text = input('Enter Plain Text: ')
    rsa(plain_text)
```

OUTPUT:



```
bhushan-borole:~/Desktop$ python rsa.py
Enter Plain Text: Bhushan
Original Message:  Bhushan
Encrypted:  129941749987299425913035
Original Message Decrypted:  Bhushan
bhushan-borole:~/Desktop$
```