Name: **Jigar Siddhpura** **SAPID:** 60004200155

DIV: C/C2 **Branch:** Computer Engineering

IS - Experiment 5 - RSA ALGORITHM

Jigar Siddhpura
60004200155
C22

Exp 5: RSA Algorithm

Aim: To study & implement RSA Algorithm

Theory:

RSA algorithm is widely used in assymmetric encryption algorithm that relies on mathematical properties of prime numbers & modular arithmetic. so,

- Select large prime no.s p & q
- Compute $n = p * q$
- Compute euler totient function $\phi(n) = (p-1)(q-1)$
- Choose an integer 'e' such that $1 < e < \phi(n)$ & e to be relatively prime with $\phi(n)$
- Determine 'd' such that $d \cdot e \equiv 1 \ (\bmod\ (\phi(n)))$
- e → public component, d → private component.
- Compute public & private keys.

- PU = (e, n), PR = (d, n)

- For encryption, $CT = (PT)^e \bmod n$

For decryption $P.T. = (C.T)^e \bmod n$

eg: p = 3, q = 11, m = 12

$n = p * q = 3 \times 11 = 33$

$\phi(n) = (p-1)(q-1) = 20$

finding e, which is relatively prime

$\therefore e = 7$

$\therefore \quad (7 \times d) \bmod 20 \equiv 1$

By extended Euclidean method

R	a	b	d	K
1	1	0	20	—
2	0	1	7	2
3	1	-2	6	1
4	-1	$\boxed{3}$	1	6

$\therefore d = 3$

For encryption,

$C.T. = (12)^7 \cdot \bmod 33$

So,

$12 \bmod 33 = 12$

$12^2 \bmod 33 = 12^4 \bmod 33 = 12$

$\therefore (12 \times 12 \times 12) \bmod 33 = 12$

For decryption,

$P.T. = 12^3 \bmod 33$

$= (12 \times 12) \bmod 33$

$= 12$

Hence proved

$\therefore$ Conclusion: So we studied RSA algorithm & implemented python code for it.

CODE

```
# Get prime numbers p and q
p = int(input("Enter p (prime number): "))
q = int(input("Enter q (prime number): "))

# Get public exponent e that is relatively prime to (p-1)*(q-1)
e = int(input(f"Enter e (relatively prime to {(p-1)*(q-1)}): "))

# Calculate n and φ(n)
n = p * q
phi = (p - 1) * (q - 1)

# Calculate private exponent d
d = 1
while True:
    if (d * e) % phi == 1:
        break
    d += 1

msg_length = int(input("Enter the length of the message (in bits): "))

print(f"p: {p}, q: {q}, message bits: {msg_length}, e: {e}")
print(f"Public key: {(e, n)}\nPrivate Key: {(d, n)}\n")

msg_data = int(input("Enter the message data: "))
print("Message data = ", msg_data)

encrypted_data = (msg_data ** e) % n
print("Encrypted data = ", encrypted_data)

decrypted_data = (encrypted_data ** d) % n
print("Original Message Sent = ", decrypted_data)
```

OUTPUT

```
PS D:\SEM-6\IS\EXPERIMENTS> python -u "d:\SEM-6\IS\EXPERIMENTS\rsa.py"
Enter p (prime number): 17
Enter q (prime number): 19
Enter e (relatively prime to 288): 5
Enter the length of the message (in bits): 8
p: 17, q: 19, message bits: 8, e: 5
Public key: (5, 323)
Private Key: (173, 323)

Enter the message data: 72
Message data =  72
Encrypted data =  21
Original Message Sent =  72
PS D:\SEM-6\IS\EXPERIMENTS>
```