Roll No:210701295

EXP NO:4
DATE:

RSA

AIM:

To implement an encryption algorithm using RSA.

ALGORITHM:

Step 1: Generate two distinct prime numbers, p and q.
Step 2: Calculate the modulus, n, by multiplying p and q: $n=p\times qn=p\times q$.
Step 3: Calculate Euler's totient function, φ(n), where $\varphi(n)=(p-1)\times(q-1)\varphi(n)=(p-1)\times(q-1)$.
Step 4: Choose an integer e such that $1<e<\varphi(n)1<e<\varphi(n)$ and ee is coprime with $\varphi(n)\varphi(n)$, i.e., gcd(e, φ(n)) = 1.
Step 5: Compute the private key, d, using the modular multiplicative inverse of e modulo $\varphi(n)\varphi(n)$, i.e., $d\times e\equiv 1(mod\varphi(n))d\times e\equiv 1(mod\varphi(n))$.
Step 6: Encrypt the plaintext message, M, using the public key (e, n), where the ciphertext, C, is calculated as $C\equiv M^e(\mathrm{mod}n)C\equiv M^e(\mathrm{mod}n)$.
Step 7: Decrypt the ciphertext, C, using the private key (d, n), where the original message, M, is recovered as $M\equiv C^d(\mathrm{mod}n)M\equiv C^d(\mathrm{mod}n)$.

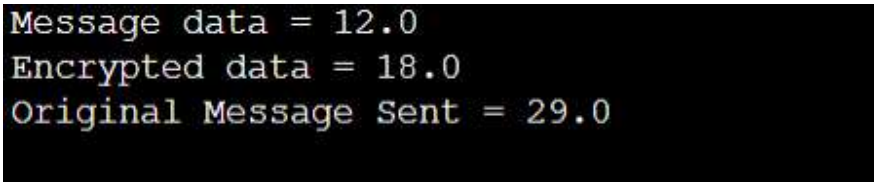
PROGRAM:

```
import java.io.*;
import java.math.*;
import java.util.*;
public class GFG {
        public static double gcd(double a, double h)
        {
                double temp;
                while (true) {
                        temp = a % h;
                        if (temp == 0)
```

```
                return h;
            a = h;
            h = temp;
        }
}
public static void main(String[] args)
{
        double p = 9;
        double q = 5;

        double n = p * q;


        double e = 2;
        double phi = (p - 1) * (q - 1);
        while (e < phi) {

            if (gcd(e, phi) == 1)
                break;
            else
                e++;
        }
        int k = 2;
        double d = (1 + (k * phi)) / e;


        double msg = 12;

        System.out.println("Message data = " + msg);


        double c = Math.pow(msg, e);
        c = c % n;
        System.out.println("Encrypted data = " + c);
```

```
            double m = Math.pow(c, d);
            m = m % n;
            System.out.println("Original Message Sent = " + m);
        }
}
```

OUTPUT:



RESULT: