EXP NO:4 **DATE:**

RSA

Aim: To implement an encryption algorithm using Rsa.

Algorithm:

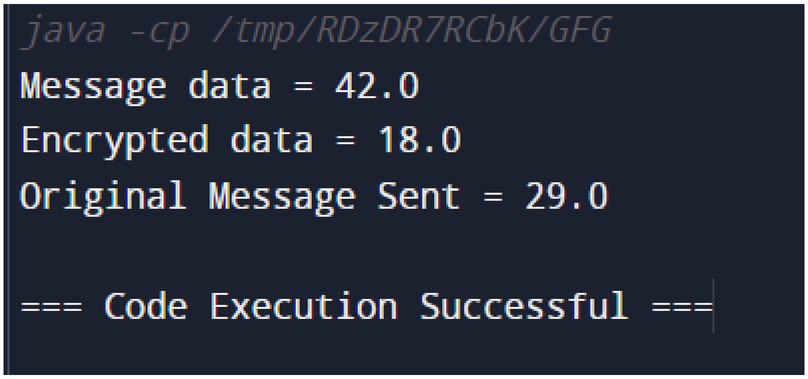
- Step 1: Select two large prime numbers, p and q. ● Step 2: Calculate the modulus, n = p * q.
- Step 3: Compute Euler's totient function, φ(n) = (p - 1) * (q - 1).
- Step 4: Choose a public exponent, e, such that 1 < e < φ(n) and gcd(e, φ(n))
  =
  1.
- Step 5: Compute the private exponent, d, such that (d * e) mod φ(n) = 1.
- Step 6: Convert the plaintext message into a numerical representation, usually using ASCII values or Unicode.
- Step 7: Encrypt the message by computing ciphertext, c, using the formula c = (msg^e) mod n.
- Step 8: Print the encrypted data.
- Step 9: Decrypt the ciphertext by computing the original message, m, using the formula m = (c^d) mod n.
- Step 10: Print the original message.
- Step 11: Return 0 for successful execution and program termination.

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 = 42;

        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:



```
java -cp /tmp/RDzDR7RCbK/GFG
Message data = 42.0
Encrypted data = 18.0
Original Message Sent = 29.0

=== Code Execution Successful ===
```

Result: