

EX.NO:

ROLL.NO: 210701288

DATE:

Implement the RSA Algorithm

AIM:

To implement RSA technique on the user input message.

ALGORITHM:

1. Select two large prime numbers p and q and compute $n = p * q$ and $\phi(n) = (p - 1) * (q - 1)$.
2. Choose a public exponent e coprime to $\phi(n)$ and calculate the private exponent d such that $d * e \equiv 1 \pmod{\phi(n)}$.
3. Convert plaintext message M to an integer and compute ciphertext $C = M^e \pmod{n}$.
4. Compute plaintext $M = C^d \pmod{n}$ using private exponent d .
5. Ensure RSA security by selecting large prime numbers and safeguarding private key d ; use RSA for secure communication, digital signatures, and encryption.

PROGRAM:

```
import java.math.*;
import java.util.*;
public class Main {
    public static int getGCD(int mod, int num) {
        // If the mod is zero, return the num
        if (mod == 0)
            return num;
        else
            // recursive function call
            return getGCD(num % mod, mod);
    }
    public static void main(String args[]) {
        int d = 0, e; // Initialization
        int message = 32; // number message
        int prime1 = 5; // 1st prime number p
        int prime2 = 7; // 2nd prime number q
        int primeMul = prime1 * prime2; // performing operations
        int primeMul1 = (prime1 - 1) * (prime2 - 1);
        System.out.println("primeMul1 is equal to : " + primeMul1 + "\n");
        for (e = 2; e < primeMul1; e++) {
            // Here e is a public key
```

```

        if (getGCD(e, primeMul1) == 1) {
            break;
        }
    }
    System.out.println("Public key e is = " + e);
    // Calculating the private key
    for (int m = 0; m <= 9; m++) {
        // get the value of temp
        int temp = 1 + (m * primeMul1);
        // private key
        if (temp % e == 0) {
            d = temp / e;
            break;
        }
    }
    System.out.println("d is : " + d);
    double cipher;
    BigInteger d_message;
    cipher = (Math.pow(message, e)) % primeMul;
    System.out.println("Cipher text is : " + cipher);

    BigInteger bigN = BigInteger.valueOf(primeMul);

    BigInteger bigC = BigDecimal.valueOf(cipher).toBigInteger();

    d_message = (bigC.pow(d)).mod(bigN);

    System.out.println("Decrypted text is : " + d_message);
}
}

```

OUTPUT:

```

primeMul1 is equal to : 24

Public key e is = 5
d is : 5
Cipher text is : 2.0
Decrypted text is : 32

...Program finished with exit code 0
Press ENTER to exit console.

```

RESULT: