Practical 4

Aim: Write a program to implement RSA algorithm.

Code:

import java.math.\*;

import java.security.\*;

public class RSA {

    SecureRandom r;

    BigInteger p, q, p1, q1, n, phi, e, d, msg, ct, pt;

    public RSA() {

        r = new SecureRandom();

        // step 1: Generate prime no. p & q

        p = new BigInteger(512, 100, r);

        q = new BigInteger(512, 100, r);

        // step 2: n=p\*q

        n = p.multiply(q);

        System.out.println("Prime no. P is:" + p.intValue());

        System.out.println("Prime no. Q is:" + q.intValue());

        System.out.println("N=P Q is:" + n.intValue());

        // step 3: Generating public key(E)

        p1 = p.subtract(new BigInteger("1"));

        q1 = q.subtract(new BigInteger("1"));

        phi = p1.multiply(q1);

        e = new BigInteger("2");

        while (phi.gcd(e).intValue() > 1 || e.compareTo(p1) != -1)

            e = e.add(new BigInteger("1"));

        System.out.println("Public key is (" + n.intValue() + ", " + e.intValue() + ")");

        // step 4:D=E^-1 mod(P-1)(Q-1)

        d = e.modInverse(phi);

        System.out.println("Private key is (" + n.intValue() + ", " + d.intValue() + ")");

        // step 5:Encryption CT=(PT)^e mod n

        msg = new BigInteger("3");

        ct = msg.modPow(e, n);

        System.out.println("Encrypted text is:" + ct.intValue());

        // step 6:Decryption PT-(CT)^d mod n

        pt = ct.modPow(d, n);

        System.out.println("Decrypted text is:" + pt.intValue());

    }

    public static void main(String args[]) {

        new RSA();

    }

}

Output:

A computer screen with white text

Description automatically generated