AIM: To implement RSA Algorithm

RSA algorithm is based on asymmetric key cryptographic algorithm. It is widely used for securing data, particularly when being sent over an insecure network.The algorithm works on prime numbers as it is extremely difficult to find factors of their products. The algorithm is as follows

1. Choose two different large prime numbers p & q.
2. Calculate n = pq. & phi = (p-1)(q-1).
3. Choose an integer *e*, 1 < e < phi, such that gcd(e, phi) = 1.
4. Compute the secret exponent *d*, 1 < d < phi, such that ed ≡ 1 (mod phi).
5. The public key is (n, e) and the private key (d, p, q). Keep all the values d, p, q and phi secret. [We prefer sometimes to write the private key as (n, d) because you need the value of n when using d. Other times we might write the key pair as ((N, e), d).]

Program

```
packagersaalgorithm;
importjava.math.BigInteger;
importjava.util.Random;
importjavax.swing.JOptionPane;
public class RSAAlgorithm
{
privateBigInteger p, q,n,n1,e,d;
private int bitlength = 1024;
private Random r;
publicRSAAlgorithm() {
      r = new Random();
          p = BigInteger.probablePrime(bitlength, r);
          q = BigInteger.probablePrime(bitlength, r);
          n = p.multiply(q);

      n1 = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
          e = BigInteger.probablePrime(bitlength / 2, r);
          while(n1.gcd(e).compareTo(BigInteger.ONE) > 0 &&e.compareTo(n1) < 0)
          {
e.add(BigInteger.ONE);
          }
                    d = e.modInverse(n1);
   }
public static void main(String[] args)  {
      // TODO code application logic here
RSAAlgorithmrsaObj = new RSAAlgorithm();
BigIntegerplain,cipher;
      String plain_text,cipher_text;
plain= new BigInteger(JOptionPane.showInputDialog("Input the string to encrypt:").getBytes()) ;
          cipher = rsaObj.encrypt(plain);
cipher_text = new String (cipher.toByteArray());
JOptionPane.showMessageDialog (null, " "+cipher_text, "Cipher Text", JOptionPane.PLAIN_MESSAGE);
plain = rsaObj.decrypt(cipher);
plain_text = new String (plain.toByteArray());
          JOptionPane.showMessageDialog (null, " " + plain_text, "Plain Text",
JOptionPane.PLAIN_MESSAGE);
   }
```

```
   // Encrypt message
publicBigInteger encrypt(BigInteger message)
   {
         returnmessage.modPow(e, n);
   }
// Decrypt message
publicBigInteger decrypt(BigInteger message)
   {
         returnmessage.modPow(d, n);
   }}
```

Output: