PROG 09

```
import java.io.DataInputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;
public class RSA
{
private BigInteger p,q,N,phi,e,d;
private int bitlength=1024;
private Random r;
public RSA()
{
r=new Random();
p=BigInteger.probablePrime(bitlength,r);
q=BigInteger.probablePrime(bitlength,r);
System.out.println("Prime number p is"+p);
System.out.println("prime number q is"+q);
N=p.multiply(q);
phi=p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
e=BigInteger.probablePrime(bitlength/2,r);
while(phi.gcd(e).compareTo(BigInteger.ONE)>0&&e.compareTo(phi)<0)
{
e.add(BigInteger.ONE); }
System.out.println("Public key is"+e);
d=e.modInverse(phi);
System.out.println("Private key is"+d);
}
public RSA(BigInteger e,BigInteger d,BigInteger N)
```

```
{
this.e=e;
this.d=d;
this.N=N;      }
public static void main(String[] args)throws IOException   {
RSA rsa=new RSA();
DataInputStream in=new DataInputStream(System.in);
String testString;
System.out.println("Enter the plain text:");
testString=in.readLine();
System.out.println("Encrypting string:"+testString);
System.out.println("string in bytes:"+bytesToString(testString.getBytes()));
byte[] encrypted=rsa.encrypt(testString.getBytes());
byte[] decrypted=rsa.decrypt(encrypted);
System.out.println("Dcrypting Bytes:"+bytesToString(decrypted));
System.out.println("Dcrypted string:"+new String(decrypted));
}
private static String bytesToString(byte[] encrypted)
{
String test=" ";
for(byte b:encrypted)  {
test+=Byte.toString(b);   }
return test;           }
public byte[]encrypt(byte[]message)       {
return(new BigInteger(message)).modPow(e,N).toByteArray();      }
public byte[]decrypt(byte[]message)        {
return(new BigInteger(message)).modPow(d,N).toByteArray();
}}
```