import java.io.\*;

import java.io.DataInputStream;

import java.io.IOException;

import java.math.BigInteger;

import java.util.Random;

public class RSAA

{

private BigInteger p;//prime number 1 private BigInteger q;// prime numbe 2 private BigInteger k;//prime numner 3

private BigInteger N;

private BigInteger NN;

private BigInteger phi;

private BigInteger e;

private BigInteger d;

private int bitlength = 1024; private Random r;

public RSAA()

{

r = new Random();

p = BigInteger.probablePrime(bitlength, r);

q = BigInteger.probablePrime(bitlength, r);

k = BigInteger.probablePrime(bitlength, r);

NN = p.multiply(q);

N=NN.multiply(k);

phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE)).multiply(k.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);

}

d = e.modInverse(phi);

}

public RSAA(BigInteger e, BigInteger d, BigInteger N)

{

this.e = e; this.d = d; this.N = N;

}

@SuppressWarnings("deprecation")

public static void main(String[] args) throws IOException

{

RSAA rsa = new RSAA();

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()));

// encrypt

byte[] encrypted = rsa.encrypt(teststring.getBytes());

// decrypt

byte[] decrypted = rsa.decrypt(encrypted);

System.out.println("Decrypting Bytes: " + bytesToString(decrypted));

System.out.println("Decrypted String: " + new String(decrypted));

}

private static String bytesToString(byte[] encrypted)

{

String test = "";

for (byte b : encrypted)

{

test += Byte.toString(b);

}

return test;

}

// Encrypt message

public byte[] encrypt(byte[] message)

{

return (new BigInteger(message)).modPow(e, N).toByteArray();

}

// Decrypt message

public byte[] decrypt(byte[] message)

{

return (new BigInteger(message)).modPow(d, N).toByteArray();

}

}