Author : Ebru Kılıç

CMPE 325 – Assignment 4 – Report

Introduction: This report is about a program that implemented RSA public- key cryptosystem.

Implementation:

About the Key Generation, I have tried to make private key high because it should not be broken easily. Therefore, I have started my code with:

```
static int bits = 1024;
```

You can see that, at the end, encrypted text is always about 309 digits.

After that I calculated the p, q, pq, N, (p-1)(q-1) values. I have explained these with comment sections on codes.

For the public key, I want system to choose random e number:

```
int min = 20;

int max = 500;

int publicKey = (int)(Math.random()*(max-min+1)+min);
```

This loop is for finding relatively prime numbers. e is a random number. If greatest common divisor is 1, it breaks but if it is not 1 it is increasing it one by one until find the prime number.

```
while (true) {

            BigInteger BigB_GCD = p1q1.gcd(new BigInteger (""+ publicKey));

      if (BigB_GCD.equals (BigInteger.ONE)) {

            break;

      }

      publicKey++;

}
```

After that there is a while loop for entering the plaintext. It is taking string character by character. Plaintext is modifiable String therefore StringBuffer has used.

After that, stringToint is for converting the string to a BigInteger. Since, string should be consist of ASCII characters only. The ASCII codes are simply concatenated to give the integer.

```
public static BigInteger stringToint(String str) {

      byte[] by = new byte[str.length()];

      for (int m = 0; by.length > m; m = m + 1)
```

Author : Ebru Kılıç

```
         by[m] = (byte)str.charAt(m);

      return new BigInteger(1,by);
   }
```

After that, intTostring is for converting the BigInteger to a string. Each byte in the integer is simply converted into the corresponding ASCII code.

```
public static String intTostring(BigInteger n) {
      byte[] by = n.toByteArray();
      StringBuffer st = new StringBuffer();
      for (int m = 0; m < by.length; m++)
         st.append((char)by[m]);
      return st.toString();
   }
```

For the decryption, decrypt has used by using the key (N,d). Each integer x in the array of integers is first decoded by computing x^d (mod N). Each decoded integers are converted into a string, and the strings are concatenated into a single string.

```
public static String decrypt(BigInteger[] cypher, BigInteger N, BigInteger prvKey)
{
      String st = "";
      for (int m = 0; cypher.length > m; m = m + 1)
         st += intTostring(cypher[m].modPow(prvKey,N));
      return st;
   }
```

For the encryption, encrypt has used by using the key (N, e). The string crumbled into chunks. After that every chunk have converted into integer. Then that integer, x, is encoded by computing x^e (mod N).

```
   public static BigInteger[] encrypt(String plain, BigInteger N, BigInteger
pubKey) {
       int chchunk = (N.bitLength()-1);
       chchunk = chchunk/8;

       while (plain.length() % chchunk != 0)
           plain += ' ';
```

Author : Ebru Kılıç

```
        int chu = plain.length()/ chchunk;
        BigInteger[] cp = new BigInteger[chu];
        for (int m = 0; m < chu; m++) {
            String st = plain.substring(chchunk*m,chchunk*(m + 1));
            cp[m] = stringToint(st);
            cp[m] = cp[m].modPow(pubKey,N);
        }
        return cp;
    }
```

Testing:

Sample Output:

```
Public key (N,e) and private key (N,d):

p value is
7504274090566537078687426086449234296301011351213193501091327147465424761290611528
977670789228584520653919403478345586699506377889144933583123830771964231

q value is
8669487563852022707551546904227077980114578057042242249749383038209577936567935200
295108961473159979842996534374610188079061275639944131760249567659579373

N = pq is
6505821090390354075651241604947597687815683311881914036194371677161993871722608249
4307139964056028347498028130434675739935699610458002753470076594220256255039814641
4544818521466792638303973768230112853258140366783552231356754845607529750124453323
53477682517281501159344416400447163621028803039168250461407163

(p-1)(q-1) is
6505821090390354075651241604947597687815683311881914036194371677161993871722608249
4307139964056028347498028130434675739935699610458002753470076594220256238866052987
0359220659077062731540851004074218770703782858376450374606727867022062457396655816
51733182020365563306388641621879510091939737695794852029863560


public key : 449 ,
6505821090390354075651241604947597687815683311881914036194371677161993871722608249
4307139964056028347498028130434675739935699610458002753470076594220256255039814641
4544818521466792638303973768230112853258140366783552231356754845607529750124453323
53477682517281501159344416400447163621028803039168250461407163

private key:
3520967761614378709094101804013955986947017917120946794644169972272526750175041881
0950189334667293738178220124043710923840478853766803271922558156782900369809467429
5094188463598499429319436066792951361427659765223780492270456284379423557121129985
33120630803894948515484275977988242655548677639372269584090969 ,
6505821090390354075651241604947597687815683311881914036194371677161993871722608249
```

Author : Ebru Kılıç

```
4307139964056028347498028130434675739935699610458002753470076594220256255039814641
4544818521466792638303973768230112853258140366783552231356754845607529750124453323
53477682517281501159344416400447163621028803039168250461407163

d value is
3520967761614378709094101804013955986947017917120946794644169972272526750175041881
0950189334667293738178220124043710923840478853766803271922558156782900369809467429
5094188463598499429319436066792951361427659765223780492270456284379423557121129985
33120630803894948515484275977988242655548677639372269584090969
```

```
Enter the plaintext, after that press enter:

     I love cryptography


RSA computed Encoded Text:


4336904729282341794139272476119239654115141992484899875526235238801867447710616264
9585327554547640332848877351004454494141186403974158245747829075291600030892427346
5730475262457792423913581357217259270017129052079559618028883133867423824634933186
64028444484634878725545547883179102757214830543717387762233537


RSA computed Decoded Text:

     I love cryptography







Enter the plaintext, after that press enter:
```

As you see, Encoded text is always around 309 digits. Compiler can convert the encoded code to decoded code same. Therefore, all the values and algorithms are correct.