CISC4080 Computer Algorithms Homework (4)[1]

[1] Distinguish yourselves, folks!

Part A: Prime Numbers (20 points)

- 1. Estimate how many prime numbers in the interval $[2^{100}, 2^{200}]$ by showing all your details.
- 2. Write the naive primality test in an algorithm format and analyze its complexity.
- 3. Implement the naive primality test such that your program at least can test following numbers

```
107
1187
2312
7091
95171
177111
8992231
20131107
```

Part B: RSA Cipher (10 points)

The following sample program: rsa_simulation.java implements a RSA cipher, where p and q are generated by using non-dterministic promaility test (Miller-Rabin test).

```
/*******************************************************
file: rsa_simulation.java
This is a sample program for CISC4080 Homework 4"
Author: Henry Han
*******************************************************/
import java.io.*;
import java.math.BigInteger;
import java.util.Random;
public class rsa_simulation {
public static void main(String[] args) throws IOException{


// 1. Generate p,q; compute n
BigInteger p=pseudo_prime(512, 100);
BigInteger q=pseudo_prime(512, 100);
BigInteger n_=p.multiply(q); //n_=p*q

// 2. Compute phi(n)=(p-1)*(q-1)
BigInteger p_minus_1=p.subtract(BigInteger.ONE);
BigInteger q_minus_1=q.subtract(BigInteger.ONE);
BigInteger phi_n=p_minus_1.multiply(q_minus_1);
System.out.println("\n this is phi(n) \n"+phi_n);

// 3. Select encrytion key e_
BigInteger e_=BigInteger.ONE;
for( int i=3;i<100;i++){
   BigInteger big_i=BigInteger.valueOf(i);
   BigInteger t=phi_n.gcd(big_i);
  if (t.equals(BigInteger.ONE)){
     e_=big_i;
     break;
   }
}
System.out.println("\nThis is encrytion key: e_\n\n" +e_);
```

```
//4. compute decrytion key d_
BigInteger d_=e_.modInverse(phi_n);
System.out.println("\nthis is the decrytion key: d_\n\n"+d_);

//5. publish the public key
System.out.println("\nThis is the public key:\n\n");
System.out.println("("+e_.toString()+","+n_.toString()+")");
// This is the message
BigInteger m=pseudo_prime(256, 100);
System.out.println("\nThis is the message:\n\n"+m);

//6. Encrpytion
BigInteger c=m.modPow(e_,n_);
System.out.println("\n"+"This is the encrypted message:\n\n"+c);

//7. decryption
BigInteger m2=c.modPow(d_, n_);
System.out.println("\nThis is the retrieved message:\n\n"+m2+"\n\n");
}

// Generate a pseudo-prime number with # of bits: 'bit_length'
// The probability that this number is prime > 1-(1/2)^certainty
   public static BigInteger pseudo_prime(int bit_length, int certainty){
            Random rnd=new Random();
            BigInteger p_prime=new BigInteger(bit_length, certainty, rnd);
            return p_prime;
    }
}
```

Implement a RSA cipher by referencing the previous codes, such that

- 1. Your p and q should be at least 1024 bits
- 2. Your RSA cipher should let user know the public key (e, n) and be able to do *at least* following *simple* encryption and decryption:

  Math is the King, Cryptology is the Queen, and We are Subjects!

  $1+$1=$1000? It is likely....:)

  AUGGCCACAUUGGCACCUCCCTTTTAAATGG

  DES is not as beautiful as RSA mathematically

  FALL CISC4080 COMPUTER ALGORITHMS

Part C: Another RSA Cipher (30 points)

Implement a RSA cipher such that

- 1. Your p and q should be prime number generated by naive primality tests
- 2. You codes should build a RSA cryptosystem and simulate its communication by the same messages you used in Part B
- 3. I will give extra credits for GUI based implementation.

Part D: Attacking RSA (20 points)

- **1.** Bob's public key is $(53, 589)$, Compute Bob's private key.
- 2. If p is public, how to compute the decryption key d ?
- 3 Bob was so proud his RSA cryptosystem: he published his public key (e, n) and $\phi(n)$ on his web, where n is a 4096 bit number. How can you attack this RSA cryptosystem based on these information, i.e., find the p and q of Bob's RSA.

What should you turn in?

1. A hardcopy of all your homework printout in class (Nov 15, 2013).
2. A folder contains all your homework assignments. If there is a programming assignment, you need to include workable source codes and related output in this folder. Please name your folder as first-name_last-name_CISC4080_homework_4. For example, John_Smith_CISC4080_homework_4 if your name is John Smith.
3. Send the zipped file (.zip instead of .rar) of your folder to xhan9@fordham.edu before 11:59 pm Nov 15, 2013.