CIS 3362 Homework #4
Number Theory, RSA
Check WebCourses for the due date

1) Determine the following values:

a) $\varphi(40950)$ b) $\varphi(84416013)$ c) $\varphi(987654321)$ d) $\varphi(79500000)$ e) $\varphi(8761077003)$

2) Without the aid of any computing device, show how one can use Euler's Theorem to determine the remainder when 45^{118874} is divided by 2014.

3) A primitive root, α, of a prime, p, is a value such that when you calculate the remainders of α, α^2, α^3, α^4 , ... , α^{p-1}, when divided by p, each number from the set {1, 2, 3, ..., p-1} shows up exactly once. Prove that a prime p has exactly $\varphi(p-1)$ primitive roots. In writing your proof, you may assume that at least one primitive root of p exists. (Normally, this is the first part of the proof.) (Note: This question is difficult, so don't feel bad if you can't figure it out.)

4) A fast way to check if a random integer x is a primitive root of a prime p is to calculate each unique prime factor q_i of p – 1, and calculate $x^{(p-1)/q_i} \bmod p$ for each q_i. If none of these are equal to 1, then x is a primitive root.

One reason for the difficulty of the discrete logarithm problem is that the sequence of numbers generated by a primitive root (α, α^2, α^3, α^4 , ... , $\alpha^{p-1} \bmod p$) doesn't seem to have any discernible pattern. If something doesn't seem to have a discernible pattern, it's a good candidate for a pseudo-random number generator.

For this question, we'll test one idea utilizing primitive roots to see if we can create an algorithm that passes the random bits tests mentioned in class.

Generate 20,000 random bits and store these in a file (whatever format you would like is fine) using the following algorithm:

1) Pick a random prime number, p, with $p > 10^6$.
2) Randomly pick integers until you identify a generator for this prime number, call this g.
3) Use a time function on your system to return a value (in C this is seconds past 1/1/1970) and then take this value mod p-1. Call this s. Repeat if s = 0.
4) Calculate $x = g^s \bmod p$.
5) Generate the 20,000 bits $(g^x \bmod p) \bmod 2$, $(g^{x+1} \bmod p) \bmod 2$, …, $(g^{x+19999} \bmod p) \bmod 2$. Basically, look at the portion of the sequence generated that starts at exponent x and for each of the next 20000 terms, look at whether or not it's even or odd.

Now, check to see if this passes the random bits tests described in class:

a) The number of 0s and 1s is in the range [9654, 10346].

b) If we break the 20000 bits into 5000 blocks of 4 bits and interpret these values from 0 – 15, letting n_i ($0 \leq i < 16$) represent the number of blocks with value i, and let

$$X = \frac{16}{5000}\left(\sum_{i=0}^{15} n_i^2\right) - 5000$$

The test is passed iff $1.03 < X < 57.4$

c) Split the 20,000 strings into maximal runs of 0s and 1s. (For example, the string 00111010111110 contains the maximal runs 00, 111, 0, 1, 0, 11111, 0, which is 4 runs of length 1, 1 run of length 2, 1 run of length 3 and 1 run of length 5.) The number of runs of each length should be in the following ranges:

Length of Run	**Required Interval**
1	2267 - 2733
2	1079 – 1421
3	502 – 748
4	223 – 402
5	90 – 223
6 or more	90 – 223

You should submit two programs for this part. One part that generates the file with the random bits and another that reads in a file with that format and determines if the bits pass the random tests or not. You should provide comments so that the grader can easily use your programs. Please use C, Python or Java.

5) Two separate RSA keys both use the same value of n = 418037. In particular, in one of the sets of keys, e = 234763 and in the other set of keys, e = 324977. It is known that the same message M has been encrypted using the public keys above yielding the ciphertexts 72801 and 323485, respectively. Determine integers x and y such that 234763x + 324977y = 1. Consequently, determine the original value of M without ever finding $\phi(n)$ or either value of d. (Hint: Remember what it means to raise a value to a negative exponent – first raise it to the -1 power, and then raise that result to the corresponding positive power. Furthermore, remember that raising a value to the -1 power means finding its modular inverse.) **Please show each step of your work. If you use one/edit one of the programs shown in class or write your own code, please include that in your write-up.**

6) Alice and Bob want to set up a new secret key using Diffie-Hellman. Their public key values are p = 12345678987654373 and g = 8888888888888888. Alice sends Bob the following value: 7153618612734331. Bob's secret number is 8234710537055521.

What does Bob send to Alice, and what is the secret key value?

7) Try to break this message. The public keys posted are 1185185228093827243 and 826535217639572223.

```
218668314887410894 786848036801024985 1077989017231680689
256764558474848789 348449708755453194 852702704314636169
```

8) What does this say?

```
ILFWCDAEPRCONDBELTDFJYXYOPSUFKMELEFTEYBTCSMLJECWALFTSPTFNHFPECLC
BDOLFTLPTHIAGIGFCE
```