Written problems

1. Suppose that m and n are integers such that $\gcd(m, n) = 1$.

   (a) Prove that if $a \in \mathbb{Z}$ is divisible by both m and n, then $mn \mid a$. (*Hint:* use Euclid's lemma).

   (b) Suppose that $a, b \in \mathbb{Z}$ satisfy the two congruences

   $$\begin{aligned} a &\equiv b \pmod{m} \\ a &\equiv b \pmod{n}. \end{aligned}$$

   Prove that $a \equiv b \pmod{mn}$ as well.

2. Textbook exercise 2.10 (same in both editions), parts (a), (b), and (c). (On a three-transmission cryptosystem)

3. Write a function that reduces the problem of breaking the cryptosystem in the previous problem to the Diffie-Hellman problem. That is, assumed that you have an efficient function `dhOracle(p,g,A,B)` that extracts the shared secret from the public parameters and transmitted values in Diffie-Hellman, and use it to write a function `analyze210(p,u,v,w)` that would efficiently find the plaintext m in the system from exercise 2.10. It is fine to write the code by hand. (Obviously I cannot autograde it because I am unwilling to confirm or deny that I have a Diffie-Hellman oracle at this time.)

   Hint. The reduction is a little tricky to find; think about all the different ways you could match up the information you know with the g, A, B from Diffie-Hellman.

4. Use the babystep-giantstep algorithm to compute each of the following discrete logarithms. Show your calculations, e.g. in the form of the table on page 83 of the textbook.

   (a) $\log_{10}[13]_{17}$ (that is, solve $10^x \equiv 13 \pmod{17}$)

   (b) $\log_{15}[16]_{37}$

   (c) $\log_{5}[72]_{97}$

5. (This problem is based on the Chinese Remainder Theorem, which we will not discuss in detail until Monday's class) Solve each system of congruences. Your answer should take the form of a single congruence of the form $x \equiv c \pmod{m}$ describing all solutions to the system.

   (a) $x \equiv 1 \pmod{3}$
   $x \equiv 2 \pmod{5}$

   (b) $x \equiv 6 \pmod{11}$
   $x \equiv 2 \pmod{10}$

   (c) $x \equiv 2 \pmod{3}$
   $x \equiv 1 \pmod{10}$
   $x \equiv 3 \pmod{7}$

   (d) $x \equiv 6 \pmod{8}$
   $x \equiv 3 \pmod{9}$
   $x \equiv 17 \pmod{17}$

Programming problems

1. Implement the Babystep-Giantstep algorithm, efficiently enough to solve the discrete logarithm problem for primes up to 40 bits in length. That is, write a function `bsgs(g,h,p)` that

finds an integer x such that $g^x \equiv h \pmod{p}$. You may assume that p is prime, $1 \leq g, h < p$, and that a solution x exists.

The test bank on Gradescope will be identical to the test bank for `disclog` on Problem Set 2, but now your code must solve all 40 cases for full credit.

(It is fine to implement any algorithm you can devise to solve the discrete logarithm problem, but BSGS is probably the easiest to implement based on what we've discussed in class.)

2. Alice and Bob use ElGamal encryption on a regular basis, using public parameters p, g. Alice's public key is A. Eve has intercepted two ciphertexts `(c11,c12), (c21,c22)` from Bob to Alice, and has determined in some way that that the plaintext of the first transmission is a specific number `m1` (e.g. it is a standard greeting).

   Furthermore, Eve has a hunch that Bob is not generating his ephemeral keys very well. After last week's problem set, he knows better than to use the same ephemeral key twice, but the keys are still related. In particular, if `k1` was Bob's ephemeral key for the first transmission, his ephemeral key for the second was computed in this way:

   ```
   k2 = u * k1 + v
   ```

   where `u,v` are two integers between 1 and 100 that Eve does not know.

   Write a function `relatedkeys(g,p,A,c11,c12,m1,c21,c22)` that Eve could use to compute the second plaintext `m2` under these assumptions. The function should return a single integer, the value `m2`. The largest test cases will use 256-bit primes p.

3. (This problem concerns the Chinese Remainder Theorem, which we will not discuss in class until Monday.) Suppose that you are given two integers m_1, m_2 with $\gcd(m_1, m_2) = 1$, and two integers a_1, a_2. Write a function `crt(a1,m1,a2,m2)` that efficiently determines the unique integer x such that

$$\begin{aligned} x &\equiv a_1 \pmod{m_1}, \\ x &\equiv a_2 \pmod{m_2}, \end{aligned}$$

   and $0 \leq x < m_1 m_2$. The fact that x exists and is unique comes from the Chinese Remainder Theorem.