Please write your hackerrank username somewhere on your problem set.
Written problems are due at the beginning of class on Friday, November 20. Programming problems must be electronically submitted by 10:50am according to the instructions on problem set 1. The submission page for the programming questions is the following.

`https://www.hackerrank.com/math158-problem-set-9/`

Written problems

1. Read example 8.10 in the textbook, about a simple *blind signature scheme* based on RSA.

   (a) Assume that the document D to be signed is a unit modulo N (we saw in PSet 8, #6 that this is extremely likely to hold). Prove that the element D' is uniformly distributed in $\mathbf{Z}/N$, i.e. that any value is just as probable as any other. This shows that Samantha cannot possibly learn any information about D from D'.

   (b) Suppose that Samantha uses the same public key (N, e) to receive encrypted messages as she uses for the blind signatures, and that Alice has intercepted a ciphertext C meant for Samantha. Show that, by requesting a blind signature, Alice can learn the plaintext corresponding to C. Explain why Samantha has no way to detect that Alice is doing this.

   Remark. This does not necessarily mean that the RSA blind signature scheme should never be used in practice (as part of a carefully designed protocol), but it does mean that Samantha should not use the same public key for both encryption and blind signing.

2. Let P be a point of order N on an elliptic curve over a fintie field. Prove that, if m, n are any two integers, the two points $m \cdot P$ and $n \cdot P$ are equal if and only if $m \equiv n \pmod{N}$.

3. (a) Consider the elliptic curve $Y^2 = X^3 + 7$. Using your code (or the solution, or someone else's code) for computing the trace of Frobenius (PSet 8, # 15), compute the number $t_p/\sqrt{p}$ (as a floating-point number) for this curve, for each prime p between 1000 and 2000, and store the results in a list. Draw a histogram of the resulting numbers (you may choose the number of bins as you see fit), and describe briefly the distibution of these values.

   (b) Do the same thing, but now for the curve $Y^2 = X^4 + 7$ (you will need to modify the trace of Frobenius code slightly since this doesn't have the form $Y^2 = X^3 + AX + B$).

   (c) Do the same thing, but now for the curve $Y^2 = X^5 + 7$. What is the main difference you notice between this distribution and the distributions you saw in parts (a) and (b)?

   Remark. The curve in part (b) is also an elliptic curve, although it's equation hasn't been expressed in our usual form. The curve in part (c) is not; it is an example of a *hyperelliptic curve*; some basics about these curves are discussed in section 8.10.

4. Textbook exercise 6.14.

5. Textbook exercise 6.16.

6. Textbook exercise 6.17.

7. Textbook exercise 6.18. *Revision:* in part (b), a short list of possible values is an acceptable answer. In practice this is just as good as getting the plaintext on the nose, since Eve can examine each possibility and see which one has the right format (e.g. is in English).

8. Suppose that Samantha has published an ECDSA verification key as described on page 322 of the textbook. Suppose that d, d' are two different documents (i.e. $d \not\equiv d' \pmod{q}$), and that Samantha signs both of them using the same random element e, resulting in signatures (s_1, s_2) and (s_1', s_2'). Prove that Eve can efficiently extract Alice's secret signing key s from these two signed documents.

Programming problems

Note. Problems 9 and 11 below will make use of public parameters specified in this document.

`http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf`

You certainly do not need to read and understand the whole document, just find the information you need for the algorithms. You'll need to look up how to convert hexadecimal strings to integers.

9. Write a program that determines whether a given ECDSA signature (s_1, s_2) for a document d is valid or not, given paramters and an ECDSA public key (notation as on page 322 of the textbook). The signatures will all use curve P-384 from the document above.

10. Suppose that Alice and Bob have performed elliptic curve Diffie-Hellman key exchange (as described as on page 317 of the textbook), but they have chosen the prime p to be only 24 bits in length. Write a program that is given the public parameters and the exchanged points Q_A and Q_B, and determines the shared secret.

    Revision: I have dropped the length of the prime from 32 bits to 24 bits, to allow a broader range of algorithms to work.

11. Write a program to decipher messages enciphered with the Menezes-Vanstone cryptosystem described in problems 6 and 7 (textbook exercises 6.17 and 6.18), given a private key and a ciphertext. The public parameters will be those of curve P-192 from the above document.

12. (*Extra credit*) Write a program to factor RSA moduli (i.e. products of two prime numbers) at least 63 bits in length. The test cases for this problem range in length from 63 bits to 128 bits, and your score is based on how many your program is able to solve, as usual.

    Solving half the test cases will be worth as many points as one ordinary programming problem, and should be possible using Lenstra's algorithm (from section 6.6). The remaining cases leave you plenty of room to try to improve the algorithm's performance, or to try any other ideas you may come up with.