MATH 470 Homework 10 (Due November 29 **on Gradescope**)

Instructions:

- FOR 5 POINT QUESTIONS, IF YOU LEAVE AN ANSWER BLANK, YOU WILL AUTOMATICALLY RECEIVE 2/5 POINTS. ANY ATTEMPTED SOLUTION WILL BE GRADED AS USUAL (AND POSSIBLY GET LESS THAN 2/5 POINTS). DOES NOT APPLY TO BONUS QUESTIONS.
- The writing of your homework submission should be done entirely on your own and you should be able to justify all of your writing in your own words.
- Show your work and write legibly. If there is any difficulty in reading or understanding your submission, or if any nontrivial steps are missing in your work, then points may be deducted.
- When you upload your submission to Gradescope, **make sure you match the correct page(s) for each question**. Otherwise your submission may not be graded, or points may be deducted.
- You may use an online calculator or code for the following operations:
  - add, subtract, multiply, quotient, remainder, rounding, non-modular square roots
  - (Extended) Euclidean Algorithm, fast powering
- You may NOT use online tools/libraries for elliptic curve operations. These should all be implemented from scratch, if you decide to use a computer.

Required problems (in the SECOND edition of textbook: "An Introduction to Mathematical Cryptography", Second Edition, by Hoffstein, Pipher, and Silverman.)

1. (5 points) 6.2(a,b) (don't do the bonus)
2. (5 points) 6.6(b)
3. (5 points) 6.9
4. (5 points) 6.11(c)
5. (5 points) 6.14(a,b)
6. (5 points) 6.21(b)
7. ("bonus" 5 points) 6.20 (use table 4.2) (note: hw is graded out of 30)
8. (More bonus up to 30 points, possibly more, depending on the complexity of your work. Email me by Dec. 5):

   Implement one of the following (you will only get points for one, even if you do multiple). **Choose your own parameters and keys to satisfy at least 128 bits of security according to NIST's recommendations** (see `https://www.keylength.com/en/4/`). You will need to generate your own random primes, find points of large order, etc. (look online for how to do these, but you should write your own code for everything EXCEPT the Schoof/SEA and hash algorithms, if you need it). If the algorithm requires a hash, use SHA-2 or SHA-3.

   (a) EC-Elgamal encryption (Table 6.6). You should read up on standard ways of encoding messages as points on an elliptic curve, and explain what you did in your implementation. Encrypt the Aggie Honor Code.

   (b) Menezes-Vanstone variant of EC-Elgamal (see exercises 6.17, 6.18). Encrypt the Aggie Honor Code.

   (c) EC-Schnorr-DSA (as described in class, or find a standard online). Sign the Aggie Honor Code.

   (d) ECDSA (Table 6.7 in textbook or look online for the more standard version with a hash). Sign the Aggie Honor Code.

   (e) Write a short expository paper on one of the following topics: Schoof's algorithm, one of the elliptic curve primality proving algorithms (see wikipdeia), or twisted Edwards curves with an explanation of EdDSA and its advantages. (or choose your own topic. Easier topics will get less points)

Suggested practice problems (do not submit): 6.5(b,d), 6.7, 6.8, 6.16, other parts of the required problems above. You should also think about the elliptic curve versions of the zero-knowledge proofs we saw based on discrete logs.