All exercise numbers from the textbook refer to the **second edition**.

1. Exercise 6.1.
2. Exercise 6.2 (Note: the notation $P \ominus Q$ means the same thing as $P \oplus (\ominus Q)$, and the notation $2P$ means the same thing as $P \oplus P$).
3. Exercise 6.5, parts (a) and (b) (*Hint.* You can save some time by making two lists in advance: values of y^2 for various y and values of $x^3 + Ax + B$ for various values of x, then checking for numbers occurring in both lists).
4. Exercise 6.6, parts (a) and (b).
5. Exercise 6.7.
6. Exercise 6.8.
7. Exercise 6.9.

Programming problems

Full formulation and submission: `https://www.hackerrank.com/m158-2016-pset-8`

8. Given a prime p, an elliptic curve over $\mathbf{F}_p$ and two points on the curve, compute the sum of the two points.
9. Given a prime p, an elliptic curve over $\mathbf{F}_p$, a point P on the curve and an integer n (up to 64 bits), compute the point $n \cdot P$ on the elliptic curve (you will want to implement the double-and-add algorithm, or something similar).
10. Given a prime p, an elliptic curve over $\mathbf{F}_p$ of order q (where q is provided for you), and two points P, Q on the curve, find the smallest positive integer n such that $n \cdot P = Q$ (i.e. solve the elliptic curve discrete logarithm problem). The value of p will be up to 28 bits long, so trial and error is unlikely to solve all of the test cases; you will most likely want to adapt the BSGS algorithm to the setting of elliptic curves.
11. Devise a method to create "blind forgeries" for a given DSA public key. That is, given p, g and A as in DSA, generate integers S_1, S_2, and D such that (S_1, S_2) is a valid signature for D. You will likely want to adapt the strategy of exercise 4.7 from Elgamal to DSA. Your method should be non-deterministic; the grading script will give the same test case multiple times to check that the same answer is not returned each time.