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

1. Exercise 4.2.
2. Exercise 4.4.
3. Exercise 4.5.
4. Exercise 4.6.
5. Exercise 4.7.
6. Exercise 4.8.
7. Exercise 4.9.
8. Consider the following implementation of one trial of Pollard's algorithm (from solutions to last week's problem set).

```
def pollard_findfact(N):
        a = random.randrange(1,N)
        # Check first whether a is a unit. If not, you have a factor.
        if fractions.gcd(a,N) != 1:
                return fractions.gcd(a,N)
        j = 2
        while fractions.gcd(a-1,N) == 1:
                a = pow(a,j,N)
                j += 1
        return fractions.gcd(a-1,N)
```

   (a) Suppose that this function is called on an input $N = pq$, a product of two distinct primes. Prove that in principle (i.e. given an unbounded amount of time), this function will always return some factor of N other than 1. Under what circumstances will it return N, rather than a proper factor?

   (b) Suppose that this function is called on $N = pq$, where both $p-1$ and $q-1$ have at least one prime factor greater than 2^{256}. Estimate how large you expect j to grow before this function will return an answer, and explain why. The result will depend on the random value of a this is chosen; try to justify why the estimate you give will be correct with very high probability.

Programming problems

Full formulation and submission: `https://www.hackerrank.com/m158-2016-pset-7`

9. Write a program that verifies whether or not a given DSA signature is valid. You will be given the public parameters and public key, and a document with a purported signature.
10. Problem 6 showed that a pair of ElGamal signatures using the same ephemeral key can accidentally give away the signer's private key. The same is true in DSA – write a program which take public paramters and a public key for DSA, along with two signed documents that have been signed with the same ephemeral key, and computes the signer's private key from this information.

11. Suppose that Samantha and Victor are using a variant of Elgamal signatures, in which the verification congruence that Victor will use is $S_1^{S_1} \cdot g^{S_2} \equiv A^D \pmod{p}$. Given the public parameters, Samantha's secret signing key, and a document D, produce a valid signature for this system.