

Problem Set 1 — Solutions

1. This is exactly as discussed in the book. I have posted the plaintexts (with punctuation and capitalization), with attribution, on the webpage.
2.
 - For the shift cipher, **Gen** outputs a random letter from $\{a, \dots, z\}$. We treat the letters as integers from 0 to 25, inclusive. Then **Enc**, on input a key k and a message $m = m_1 \cdots m_\ell$ (where each m_i represents a character) outputs the ciphertext $c = c_1 \cdots c_\ell$ where $c_i := [(m_i + k) \bmod 26]$. Decryption algorithm **Dec**, on input a key k and a ciphertext $c = c_1 \cdots c_\ell$ (where, again, each c_i is a character) outputs $m = m_1 \cdots m_\ell$ where $m_i := [(c_i - k) \bmod 26]$.
 - For the substitution cipher, **Gen** outputs a random permutation of $\{a, \dots, z\}$. This can be represented as an array $k = k[0], \dots, k[25]$ where each $k[i]$ is a different letter of the alphabet. Again, treating characters as integers in the range $\{0, \dots, 25\}$, the encryption algorithm **Enc** does the following: on input a key k , as above, and a message $m = m_1 \cdots m_\ell$ (where each m_i is a character), output the ciphertext $c = c_1 \cdots c_\ell$ where $c_i := k[m_i]$. The decryption algorithm **Dec**, on input a key k and a ciphertext $c = c_1 \cdots c_\ell$, output the message $m = m_1 \cdots m_\ell$ where each m_i is the unique value such that $c_i = k[m_i]$.
 - For the Vigenere cipher (assuming a key length of 5), **Gen** chooses random $k_i \in \{a, \dots, z\}$ for $i = 1$ to 5, and outputs the key $k = k_1 k_2 k_3 k_4 k_5$. Encryption of a message $m = m_1 \cdots m_\ell$ using the key k returns a ciphertext $c = c_1 \cdots c_\ell$ where $c_i := [(m_i + k_{[i \bmod 5]}) \bmod 26]$. Decryption of the ciphertext $c = c_1 \cdots c_\ell$ using the key $k = k_1 k_2 k_3 k_4 k_5$ returns the message $m = m_1 \cdots m_\ell$ where $m_i := [(c_i - k_{[i \bmod 5]}) \bmod 26]$.

In all cases, the message space is $\{a, \dots, z\}^+$, i.e., strings of non-zero length over the English-letter alphabet.

3. Define **Enc** as follows (defining **Dec** is left to the reader):

$\text{Enc}_r(m)$
Compute $k := \text{Gen}'(r)$
Output $c \leftarrow \text{Enc}'_k(m)$

(Note that Gen' as written is deterministic since we are explicitly fixing its random coins r ; the encryption algorithm Enc' may be randomized.)

To see that this produces ciphertexts with the right distribution, consider the distribution of the key k generated in the first line of **Enc**. Here, $k := \text{Gen}'(r)$ for a uniformly-random value of r . (The coins r are uniformly random since, by definition, this is what **Gen** outputs). This is identical to the distribution of keys generated

by Gen' : In each case, k is generated by choosing random r and running Gen' ; the only difference is that in the current scheme Enc that is running Gen' as a subroutine, whereas in the original scheme Gen' itself was used to generate k .

In any event, this means that, in both schemes, Enc' is run using the same distribution on keys. So Enc will produce ciphertexts with the same distribution as Enc' .