Homework 1

1 Problem 1 (20 points)

Consider the following encryption scheme. The key will be a table such as

	0	1	2	3	4	5	6	7	8	9
	g	z	a		t	k	f	w		s
3	c	m	e	.	b	x	p	u	h	y
8	i	d	v	r	_	q	j	l	n	o

The plaintext alphabet will consist of the 26 letters, plus spaces (represented by _) and periods. In general, the table will have the following form. It will consist of 4 rows. The numbers 0-9 are always written in the first row, in increasing order. We will call these the column indices. Then 8 plaintext characters are placed in the first row, leaving two black spot. The numbers from the first row corresponding to those blank spots (in our example, 3 and 8) are then written in the third and forth row of the first column, again in increasing order. These will be called the row indices. The remaining 20 characters fill out the rest of the third and fourth row.

Encryption is character by character. For each character, find the character in the table. There are two cases:

- If the character is in the second row (the first row of non-numbers), that character is encrypted by its column index. So, for example, "a" becomes "2".
- If the character is in the third or fourth row, that character is encrypted by its row index followed by its column index. So "x" gets encrypted as "35".

The overall ciphertext is just the concatenation of the encryptions of each letter. So for example, "attack." is encrypted to "2 4 4 2 30 5 33". The spaces between the numbers are just to show how the various letters map to numbers; in reality the actually ciphertext would be "244230533" with no spaces.

(a) Explain how to decrypt. Explain for general keys, not just the key given above.

(b) How many possible keys are there?

(c) Explain why vanilla frequency analysis wont work. What about frequency analysis using digrams? What are the features of this scheme that lead to this conclusion? You do not need to prove anything; just an intuitive explanation.

(d) A friend suggests putting characters such as "a", "e", "i", "o", "r", "t" and maybe "_" in the second row. What *non-security* reasons might your friend have for this suggestion? Do you think this suggestion will help or hurt security, compared to a random placement of characters. (This problem is meant to be open ended)

2 Problem 2 (20 points)

Sometimes data is lost or corrupted in transit. Suppose Alice encrypts a message m, and transmits the ciphertext c to Bob. However, in route to Bob, the ith ciphertext character gets deleted, and Bob only receives c' which is c with the ith character deleted. Bob does not know which character was deleted, or even that any character was deleted. When Bob decrypts, he will then get a different message m'. Hopefully, m' is not to different from m, so that Bob still learns the bulk of the desired message.

(a) In the case of a simple substitution cipher, what is the impact of deleting a single ciphertext character on m'? Explain.

(b) What about polygraphic substitution? Explain.

(c) What about the Vigenère cipher? Explain.

(d) Let $\Sigma = \{a, b, c, \ldots, z\}$. Let $\mathcal{M} = \Sigma^\ell$ and $\mathcal{C} = \Sigma^n$ for some integers ℓ, n. Suppose you have an encryption scheme $\mathsf{Enc}, \mathsf{Dec}$ with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ that is guaranteed to be correct. Let $\mathsf{Enc}_2(k, m)$ do the following encryption algorithm with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}_2 = \Sigma^{2n}$: first compute $c = \mathsf{Enc}(k, c)$, and then output c_2, which is c, except that each bit is repeated twice. So if $c = 01001$, then $c_2 = 0011000011$.

Devise a decryption algorithm Dec_2 with the following guarantee. If $c_2 = \mathsf{Enc}_2(k, m)$ and c_2' is obtained from c_2 by deleting any single character, then $\mathsf{Dec}_2(k, c_2') = m$. This must hold regardless of the choice of m or the position of the deleted character. Your Dec_2 will run Dec as a subroutine. You do not need to analyze the security of $(\mathsf{Enc}_2, \mathsf{Dec}_2)$.

(e) Suppose instead we tried the following transformation: $\mathsf{Enc}_3(k, m) = \mathsf{Enc}(k, m_2)$. Here, m_2 is m, but with each bit repeated twice (So if $m = 01001$, then $m_2 = 0011000011$). Enc_3 has message space $\Sigma^{\ell/2}$ (we assume ℓ is even) and ciphertext space Σ^n.

Show that this transformation does *not* in general achieve the desired property. To do so, you will give a concrete example of an encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$. To keep things simple, your scheme $(\mathsf{Enc}, \mathsf{Dec})$ does not actually need to be secure, but it must be correct. Then you will plug $(\mathsf{Enc}, \mathsf{Dec})$ into the conversion above to obtain Enc_3. Now Alice chooses a uniformly random message $m \in \Sigma^{\ell/2}$, computes $c_3 = \mathsf{Enc}_3(k, m)$, and then Bob receives c_3', which is c_3 with a the ith character deleted, for some i. You must show that there are some i such that Bob will be unable to compute m with certainty, no matter what decryption algorithm he may be using. (Bob gets to know the key k).

3 Problem 3 (20 points)

(a) Show that a Homophonic substitution cipher that encrypts a single character at a time can *never* have perfect secrecy, even for two-character messages, no matter how large the output alphabet is. That is, provide two different two-character messages such that the distributions of encryptions of those two messages are different, regardless of the ciphertext alphabet size or the exact mappings between characters.

(b) Show that the Vigenère cipher does *not* have perfect secrecy if the message length is even one character longer than the key length.

(c) Show that the Vigenère cipher does *not* have perfect secrecy if the keyspace is grammatically correct English, even if the message length is smaller than the key length.

(d) Show that the following cipher has perfect secrecy. Messages are ℓ bit strings. The key is a random permutation P on 2ℓ items. To encrypt a message m, write down m, followed by $\overline{m}$, the bitwise complement of m. Then permute the bits of the resulting 2ℓ-bit string $m||\overline{m}$ according to the permutation described by the key. To decrypt, perform the inverse permutation to recover $m||\overline{m}$ and discard $\overline{m}$.

(e) What happens in part (d) we replace $\overline{m}$ with 0^ℓ (here, 0^ℓ denotes a sequence of ℓ zeros). That is, we apply the permutation P to the 2ℓ-bit string $m||0^\ell$. Is the scheme still correct? Is it still secure? Prove why or why not.

4 Problem 4 (20 points)

(a) Devise an encryption scheme such that (1) given an encryption of any message, an adversary can figure out 90% of the secret key, but (2) the scheme is still

perfectly secure, despite 90% of the key being revealed. Do not forget to prove that the scheme is secure and that it is correct.

(b) Devise an encryption scheme such that (1) given an encryption of any message, an adversary learns *nothing* about the secret key, but (2) the scheme is completely broken (as in, given the ciphertext, an adversary can completely recover the plaintext).