CRYPTOGRAPHY

LAMC HIGH SCHOOL 1

1. Historical Cryptography

For almost as long as humans have been sending messages, they have been trying to keep messages secret. Fairly quickly, we discovered that encrypting information was important. A good method of encryption (called an *encryption scheme*) should make it:

(1) easy to **encrypt** the message - converting the original into the version sent.
(2) easy to **transmit** the message - it needs to reach the recipient quickly.
(3) easy to **decode** the message - the intended recipientt should be able to read it quickly.
(4) hard to **decipher** the message - someone intercepting the message should have a lot of trouble reading it.

Each of these criteria are all relative to the tools available, and some have changed greatly over time. Some of the methods used today make encryption and decryption very easy, even though applying them in the time before computers would have been infeasible. Additionally, one of the first uses of computers was to decipher codes that proved resilient to cracking by hand.

Problem 1: for each of the following encryption schemes, say which of the above criteria they are good at and bad at.

- shaving someone's head, writing a message on it, and letting their hair regrow before sending them to deliver it.
- for each letter in the message, flip a coin: on heads, delete it.
- write in a different alphabet.
- change each letter to a different one in a pre-determined manner.

Early encryption methods centered around hiding the actual existence of the message - the head-shaving scheme is said to have been employed in ancient Greece. As cool as some of them were, there isn't much math to them. the first mathematical encryption scheme was the Caesar cipher:

Definition: a Caesar cipher encrypts a message by choosing a number from 1 to 25, and shifting each letter of the message forward by that many (with a shift of 1 from Z taking you back to a).

this relies at its heart on the idea of modular arithmetic: each letter can be represented as a number from 0 to 25, and Caesar shifting is adding a fixed number, mod 26.

Problem 2

- encrypt the message "MATH" with a Caesar cipher with 4 as the key.
- the message SKKZ NKXK was encrypted using a Caesar cipher. decipher it.
- Your friend wants to send you a message encrypted with a Caesar cipher. they're worried, though, because there are only 26 such ciphers (and one of them leaves the message untouched). so, your friend chooses 100 different Caesar ciphers, and encrypts the message first with the first cipher, then encrypts that new message with the second cipher, and so on, 100 times. they reason that there are 26^100 ways to choose 100 Caesar ciphers, so this is far more secure than a single Caesar cipher. Is your friend correct? Why or why not?

2. affine ciphers

While it is a good introductory cipher, being able to be cracked within 25 attempts every time renders the Caesar cipher impractical for real encryption. Here is a way to generalize it:

Definition: An **affine cipher** is an encryption scheme in which the key is a pair of numbers $a, b \in \{0, 1, \ldots, 25\}$. The affine cypher with key (a, b) sends the number n to $an + b \pmod{26}$.

Problem 3 For a cipher to be decodable, no identical messages can be encrypted to the same message. For which keys (a, b) is the affine cipher decodeable?

Problem 4 Consider an alphabet with p letters, with p prime. How many decodable affine ciphers are there for that alphabet? What if the alphabet has p^2 letters? What if it has pq letters, with p and q different primes?

This is definitely better than the Caesar cipher, but in the age of computers, it's still a pretty small number of potential encodings. Ultimately, though, any encryption scheme that relies on using a permutation of the alphabet is vulnerable to *frequency analysis*: some letters are used more often, so with a big enough message, you can use your knowledge of the language to deduce the encryption (there are even published puzzles that only consist of this: they're called cryptograms).

Problem 5 Decrypt this profitable message, which was encrypted with an affine cypher: Yqb gvoty ebotzu yz ybkk Wzu yqb rzkzo zg yqb tfx lbyt h wzkkho.

since any permutation of letters is vulnerable to frequency analysis - in a long enough message (like this handout), whatever you're sending E to will be the most common letter, etc. - to be truly secure, we can't statically change letters, we need to change each letter in a different way.

Definition: A **One Time Pad Cipher** is an encryption scheme that has a randomly generated key at least as long as the message; to encode a message, you add the message to the values of the key mod 26, and to decode, subtract mod 26.

The original ciphers of this type didn't use randomly generated keys; they used books, like the bible, or Shakespeare plays. However, if the key text was guessed, the code could be broken, leading to the desire for absolute randomness.

Problem 6 How could one decipher a true one-time pad cipher?

In a certain sense, a one-time pad cipher is the best possible encryption scheme - without knowing some secret information (the hidden key), it's literally impossible to decipher a message - because the key is random, it's equally likely to be any message of the same length (and you can hide the length of a message by padding the end with extra characters).

Problem 7 It can be very impractical to carry around a one-time pad; it has to be quite long to be effective. So, here is a proposed alternative to true randomness: pick a number $a_1 \in \{1, 2, \ldots, 25\}$, and let $a_k = (a_{k-1}^2 \bmod 26)$. Shift the k^{th} letter of the text by a_k. Now, instead of keeping a huge document, we can just remember a single value; is this a good encryption scheme? Why or why not?

4. Public Key Encryption

Ultimately, if you need to transmit encrypted information, you'll need to transmit information about the encryption itself (or no one will be able to read you), so encryption schemes that allow you to do so without letting everyone crack the code are desirable. To do this, we'll need something (preferably, something mathematical) that is easy to do, but hard to undo - and the current most popular choice, where the decider of easy and hard is computational power, is multiplication/factoring:

Problem 8 Using long multiplication to multiply an n-digit number by an m-digit number, how many steps do you need? A step is multiplying one digit by another, or adding a digit to another.

Problem 9 Long multiplication isn't actually the fastest way to multiply two numbers; here is a faster algorithm to:

- If we are multiplying two-digit numbers x_1x_2 and y_1y_2, first compute $A = x_1 \times y_1$ and $B = x_2 \times y_2$
- Compute $C = (x_1 + x_2) \times (y_1 + y_2)$
- Compute $K = C - A - B$; what is this equal to in terms of $x_1, x_2, y_1, and y_2$.
- Compute $100A + 10K + B$

Explain how to generalize this to longer numbers. About how many steps will it take (in terms of the lengths of the numbers)?

Problem 10 How many steps does it take for you to factor an arbitrary number (as a function of its length)?