Homework 1

Encryption

Botond Lajos Perényi

Task 1: Affine cipher in $\mathbb{Z}_{97}$.

(a) Encryption and Decryption functions with key $k = (a, b)$ (the message is m and the resulting ciphertext is c):

$$E(k, m) = a \cdot m + b \bmod 97$$
$$D(k, c) = a^{-1} \cdot (c - b) \bmod 97$$

(b) The number of possible meaningful and different keys is

$$|a| \cdot |b|$$

Where $|a|$ is the possibilities for a and $|b|$ is the different possibilities for b. b can have 97 different values, and a can have as many as there are co-primes below 97 with 97. This is given by Euler's Totient function: $\phi(97) = 96$ (this is because 97 is prime).
In total the number of different keys is:

$$96 \cdot 97 = 9312$$

(c) We have an intercepted ciphertext-plaintext pair: (DOG)=(28 83 43), and a ciphertext with the same key: (78 23 33).
To figure out the key we can use two letters from the known ciphertext-plaintext pair with either the Encryption or the Decription functions: (I choose the decryption with letters (DO) = (28 83))

$$D(k, 28) = a^{-1} \cdot (28 - b) \bmod 97 = (D)3$$
$$D(k, 83) = a^{-1} \cdot (83 - b) \bmod 97 = (O)14$$

Thus we get:

$$a^{-1} \cdot (28 - b) \bmod 97 = 3$$
$$a^{-1} \cdot (83 - b) \bmod 97 = 14$$

Rewriting the equations we get:

$$b = 28 - 3a \bmod 97$$
$$b = 83 - 14a \bmod 97$$

Making the two equal each other:

$$28 - 3a = 83 - 14a \bmod 97$$

With rearranging we get:

$$55 = 11a \bmod 97$$

We need the multiplicative inverse of 11 in mod97, which is 53. (Using the Extended Euclidean Algorithm this is fairly straightforward.) With multiplication we get:

$$a = 55 \cdot 53 \bmod 97 = 2915 \bmod 97 = 5$$

Substituting back to one previous equation we get:

$$b = 28 - 3 \cdot 5 \bmod 97 = 28 - 15 \bmod 97 = 13$$

Meaning the key is: $k = (5, 13)$
For decrypting the second message we firstly need the multiplicative inverse of $5 \pmod 9 7$ which is 39. Writing this into the decryption function we get:

$$D(k, 78) = 39 \cdot (78 - 13) \bmod 97 = 13(N)$$
$$D(k, 23) = 39 \cdot (23 - 13) \bmod 97 = 2(C)$$
$$D(k, 33) = 39 \cdot (33 - 13) \bmod 97 = 4(E)$$

The encrypted message is (NCE).

Task 2 Histograms for different ciphers
My choice of English plaintext is from "The Hobbit":
In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort
This is the original frequency analysis of the text:

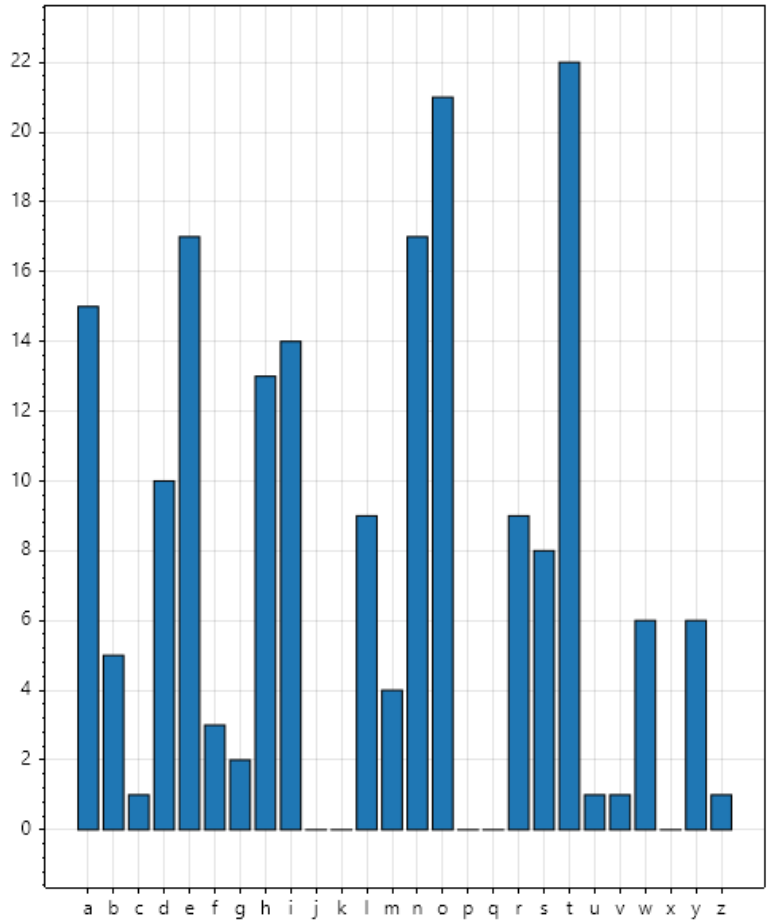

Figure 1: The plaintext histogram

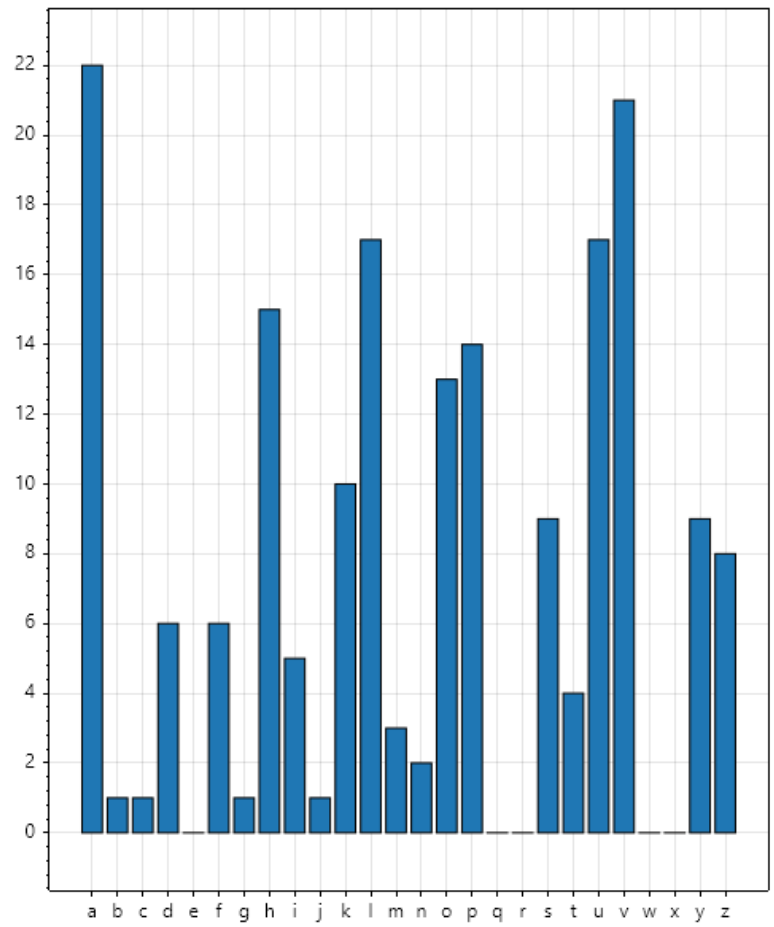

Figure 2: Ciphertext histogram after shift cipher

Then I generated a ciphertext using shift cipher with key $k = 7$.
On the histogram 2 the bars' height does not change compared to the original text, only their position moves with the key. This is the key giveaway for the shift cipher.

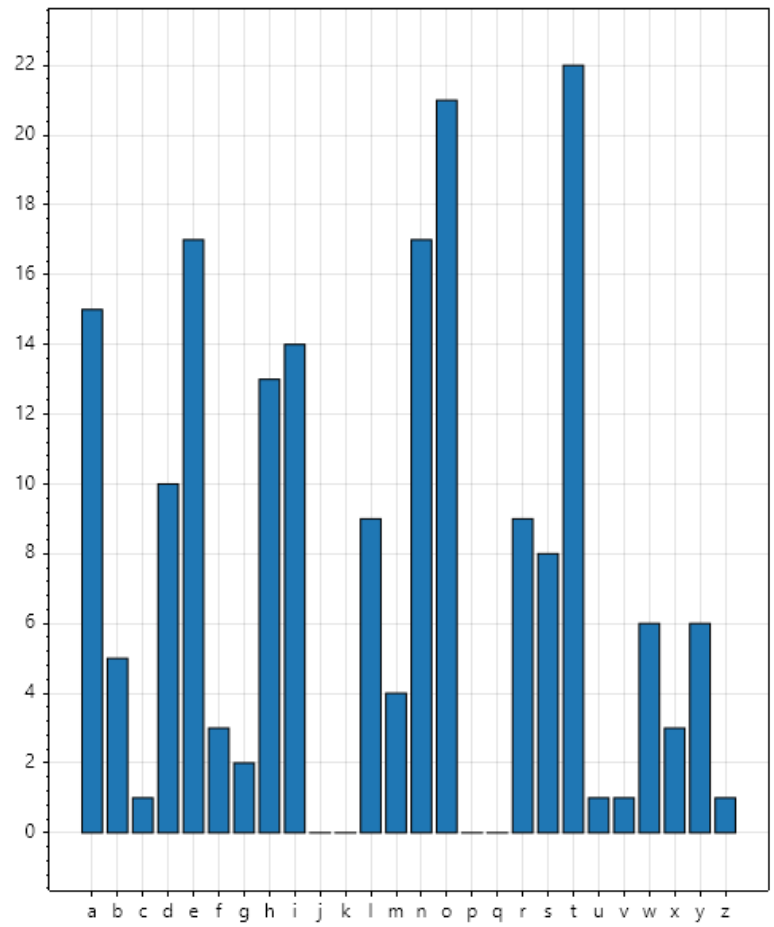

Figure 3: Ciphertext histogram after permutation cipher

Then I generated a ciphertext with permutation cipher using the key $(7, 2, 5, 3, 8, 4, 1, 6)$, meaning the length of the key is 8. I denoted the first element with 1 in the key.
The histogram 3 shows that all the letter frequencies remained the same, except for x, which I used to pad out the last segment of the plaintext. This is the main property for permutation ciphers, the letter frequencies do not change, only the individual letters' position.

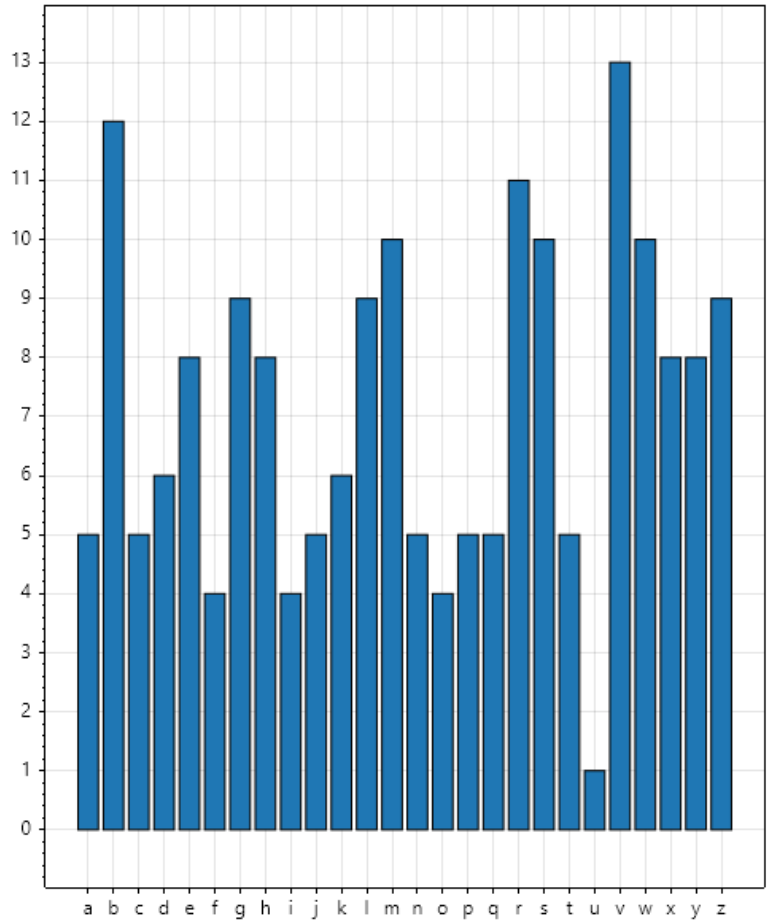

Figure 4: Ciphertext histogram after Vigenère cipher

The last ciphertext is generated with Vigenère cipher, using the key (*tolkien*) (I felt this to be appropriate for the plaintext).
The resulting histogram 4 shows a much more evenly distributed letter frequency. This is the result of each individual letter being encoded with a different individual key. The more even histogram indicates a Vigenère cipher.
For each cipher I wrote the code using C#. The histograms are generated using ScottPlot, which is a free tool for .NET. The code is available (publicly) on GitHub. Program code

Task 3 What cipher would be better for Iulius Caesar?
Assume that explaining and using all the historical ciphers would be possible on a level, that an average ancient Roman can understand. (This would be a big limitation, because otherwise one can only use a cipher that is understandable for those who need to operate it.) In this case the best cipher would be the One-Time Pad, because it is perfectly secure. However, operating such cipher requires an already established prior communication over a secure channel, in which case the encrypting is pointless.
The second best option (in my opinion) is to use Vigenère cipher with a password that is comparable in size to the average message. (I assume that the messages are fairly short, based on the application.) This can achieve OTP level security with one message (multiple intercepted messages can be broken with sufficient effort).
All other ciphers that we discussed are more easier to break with statistical and brute force methods.

Task 4 Breaking One Time Pad

One Time Pad is unbreakable. With only the intercepted ciphertext I can not make any guess on the key or the plaintext. Every guess would be as likely as the next one. The only thing that can help finding a more plausible plaintext would be the statistical likelihood of one 3-letter word to show up randomly.

Task 5 Breaking ciphertext encrypted with Vigenère cipher
The first task is to find recurring letter groups, which indicate that the same letters were encoded with the same key (or part of the key). For this I used my code focusing on groups, that have at least 3 length (anything smaller than that could be random noise, so I discarded that). Then I selected a few candidates that I found, and located them in the ciphertext (I used my code which returned their zero based positions). I got the following (with the differences between consecutive appearances, which I need later):

- `FHK` in positions 0, 20, 55, 185 (20, 35, 130)
- `VKO` in positions 118, 339, 409 (221, 70)
- `GQG` in positions 303, 328, 373 (25, 45)

The difference between two appearance of a group of letters (trio of letters in this case) is most likely a multiple of the key, based on how the Vigenère cipher works. In our case this is most likely 5. (The greatest common divisor among the position differences if we exclude 221.)
To determine if this assumption is correct I calculate the index of coincidences for each part of the ciphertext. (A part is every fifth letter of the ciphertext from the first, second, ... fifth letter.) For this I use my own code (Program code).
These are my results for the index of coincidence calculations:

- 0.06224899598393574
- 0.060527825588066554
- 0.05823293172690763
- 0.05789009697325889
- 0.05524537173082574

These are all around (just somewhat below) the expected value for meaningful text, 0.065. This reinforces the idea that the key's length is indeed 5.

For each task I used the materials presented in the lectures and my code that is available on Github as it is at the time of writing (March 4, 2025): Program code.