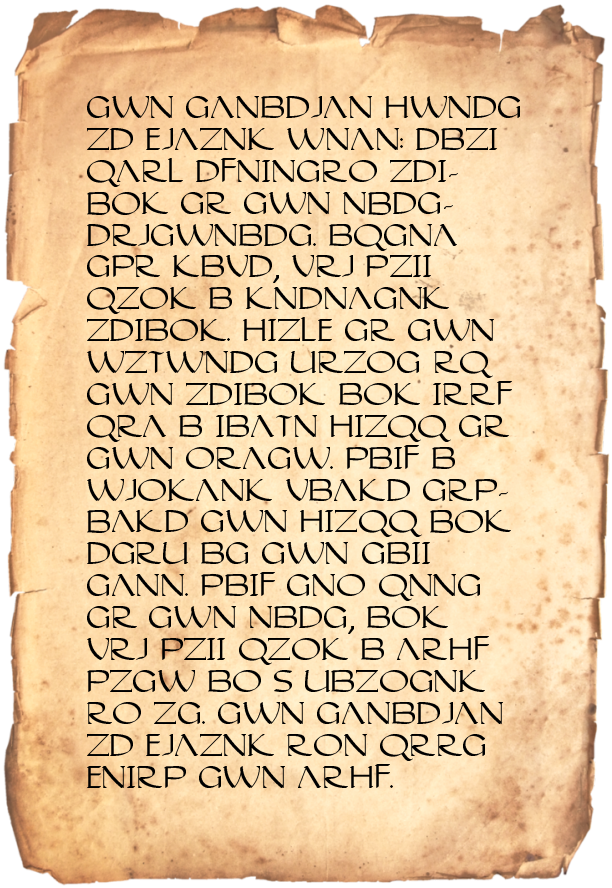
Web Security
Homework Assignment 1

COSC 4371

2018 Spring

Problem 1: Treasure Hunt (4 points) You have come across a mysterious old document, which is rumored to be the key to finding the fabled treasure of the pirate Captain Blackbeard (worth 4 points). Decrypt the text to learn the location of the treasure chest (and to successfully solve the problem)!



Hints:

- Most 17th century pirates have not taken any web security classes and, thus, have not learnt that *affine ciphers* can easily be broken.
- The plaintext was written in contemporary English. Note that only the letters of the alphabet (A, B, ..., Z) are encrypted, punctuation marks and spaces are not.

Tasks:

1. Write a Java function to calculate the frequencies of letters in the ciphertext, and find at least two pairs of corresponding plain and cipher characters! (2 points)
2. Express the decryption of a character as an affine transformation in modulo 26 arithmetic, and complete the source code to decrypt the ciphertext! (2 point)

Note that in Java, `-9 % 26 == -9` and `-9 % 26 != 17`.

Problem 2: `Robot Mayhem` **(4 points)** Computers have become self-aware, and they are trying to take over the world! Luckily, the human resistance was able to send a lone cryptanalyst, you, back in time to save humanity (and solve Problem 2 for 4 points). To stop the machines, you have to decrypt the following ciphertext and retrieve the secret password, which can be used to shut down the self-aware computers.

```
-119, 119, 48, -18, 29, 23, -85, 81, 22, -85, 70, 74, -66, 90,
20, -15, 66, 5, -67, 65, 19, -95, 64, 0, -13, 83, 5, -68, 86,
18, -81, 64, 15, -18, 122, 48, -102, 98, 75, -1, 28, 85, -60
```

The following is known about the ciphertext:

- The plaintext is an `HTTP GET` request encoded in ASCII.[1]
- Each number in the ciphertext above is a signed byte (i.e., Java `byte` type).
- The algorithm that was used to encrypt the plaintext is binary "many-time pad."
- The key consists of an unknown (but not too high) number of bytes.

Tasks:

1. Determine the length of the key using a brute-force approach: for each key length (i.e., 1, 2, 3, ...), try to recover the key using your knowledge of the plaintext! (2 points)
2. Complete the source code to decrypt the ciphertext! (2 points)

Note that in Java, the XOR (i.e., modulo 2 addition) operation can be performed as
`byte xor = (byte)(byte1 ^ byte2);`

[1] Note that this is the same as UTF-8. In Java, you can covert a `byte` to a character simply using `(char) myByte`, and convert a character to a byte using `(byte) myChar`.