The Caesar Cipher

Reyansh Tellis

1 Section A: General Information and Mathematics

1.1 Overview

The Caesar Cipher, named after and theorised to have been used by Julius Caesar, is a type of monoalphabetic shift cipher in which each letter in the plaintext is replaced by another letter some fixed number of positions down the alphabet. For example, with a left shift of 3, the letter D is replaced by A, the letter E is replaced by B and so on. Therefore, a basic encryption formula for a letter x by shift n can be mathematically described as:

$$E_n(x) = (x + n) \bmod 26 \tag{1}$$

with decryption being performed similarly:

$$D_n(x) = (x - n) \bmod 26 \tag{2}$$

There are only 25 possible shifts values: from $1 \rightarrow 25$ inclusive, as the values of 0 and 26 yield no difference to the original ciphertext. Thus, the Caesar cipher is widely considered to be one of the easiest methods to encipher text messages with. Due to its extremely limited *keyspace*, the Caesar cipher is highly vulnerable to brute-force attacks from computers, where all 25 shifts can be sequentially tried and analysed with *Monogram Frequency Analysis* to easily determine the plaintext. This is the core idea behind my code, explained in **Section B**.

1.2 Example

The table below demonstrates a shift of 7. i.e. $A \rightarrow H$ etc. Tables such as these can be used to encrypt and decrypt messages. For example, the following ciphertext can easily be decrypted through the table to yield the plaintext.

Original	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher	H	I	J	J	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G

Max Vtxltk Vbiaxk ptl gtfxw tymxk Cnebnl Vtxltk

The Caesar Cipher was named after Julius Caesar

2 Section B: My Code

2.1 Overview and Runtime

The core idea behind my code is using brute-force to sequentially try all 25 shifts and then determine, using monogram analysis, the correct plaintext. The code is split into 3 main functions which are further explored in subsection **2.2**.

The average runtime of the program was 27.8ms ± 188µs, for a specific, average-sized ciphertext. A larger ciphertext should not drastically alter the runtime. These runtimes are proof of the Caesar cipher being vulnerable to such attacks, as all the analysis, calculations and shifts are performed in only 2 hundredths of a second.

2.2 Exploring Individual Functions

i. **Caesar Converter:** This function takes in two inputs: the *ciphertext* and the *shift* and uses the decryption algorithm to return a string of values corresponding to this particular shift. Numerical values of letters are calculated using indexes of the *alpha* list

ii. **Calculating Most Probable:** This is the main function of the program. It iterates from $1 \rightarrow 25$ and calls the **Caesar Converter** function each time in order to keep track of each decryption variation by adding them to the *plaintexts* list. It then uses monogram frequency analysis to determine optimal shift; in other words, the shift which most closely matches the standard letter distributions for letters in the English Alphabet. Finally, it returns the correct plaintext, based on the calculated best shift.

iii. **Main:** The *Main* function is the function which handles user input and displaying of results, as well as giving the user an option to try new shift values for further manual decryption.