**CAESAR’S CIPHER:**

**TASK 1**

What is the keyspace for the Caesar Cipher. Is it possible to implement a brute force attack on this cipher or not?

* The key space of the Caesar cipher is relatively small , 25 integers number{1,2,3..25}.So, in the case of a brute force attack the best scenario case is to decipher the given ciphertext after the first iteration and the worst case scenario is after 26 consecutive attempts.

**TASK 2**

Implement this version of Caesar’s cipher (both encryption and decryption). First, a user should be

prompted to enter the shift number K (how would you solve the problem of an invalid K). Subsequently, the user

should be given the option to decrypt or encrypt a message. The user should be prompted to input the plaintext

(or the ciphertext – depending on the selected mode) to be encrypted or decrypted. The program, on successful

run, should output a valid ciphertext or plaintext depending on the selected operation mode.

* The implementation of the programme is on the **tutorial\_1\_ex1\_part\_1.py** file.

• What is considered to be an invalid K?

Any value which is not an integer, and any integer K values which produce the number 0 after calculating the Kmod26, e.g. 0, 26, 42, 68.

Implement a simple timer to measure the encryption and decryption times. What are your observations?

* The implementation of the programme is on the **tutorial\_1\_ex1\_part\_1.py** file.

On average decryption is slightly faster than encryption. In some cases, even a lot faster after decoding the produced ciphertext as showcased below

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, απόδειξη

Περιγραφή που δημιουργήθηκε αυτόματα   
  
 **TASK 3:**TASK 3: In this task, you are to break the Caesar’s cipher you implemented in Task 2. To do so, you are required

to write a program that will do the following:

• Read the ciphertext "IYE RKFO NYXO GOVV DY VOKBX DRSC DOMRXSAEO. LED DRSXQ GSVV QOD

WYBO NSPPSMEVD - KNWSX";

• Calculate all possible keys for Caesar’s cipher;

• Using each calculated key, attempt to decrypt the given ciphertext;

• Output all possible keys along with the corresponding decrypted message. What is the original message?

• How long does your program take to recover the encryption key and the original message?

Ηas been implemented on the 2nd script.  
  
Original message is:  
  
On the 11th iteration(K=10) : YOU HAVE DONE WELL TO LEARN THIS TECHNIQUE. BUT THING WILL GET MORE DIFFICULT - ADMIN | Time elapsed 0.29500000000000004 milliseconds

**Double Caesar’s cipher:**

Using the program you created in Exercise 1: Task 2, implement the Double Caesar’s cipher described above (both encryption and decryption);

* Implemented on the third script,

2. Decrypt the ciphertext "PDA BENOP NQHA KB BECDP YHQX EO UKQ ZKJ’P PWHG WXKQP BECDP YHQX".

In which movie was the plaintext used?

- Fight Club

3. Is it more secure? If so, is it considered fully secure?

No, it as secure as the single Caesar’s cipher due to the reason that the key space is not increased and practically using a brute force algorithm would allow us to decipher the cipher text with the same ease.

4. What can you generally observe by using this double encryption?

Using double Caesar is equivalent to using a single Caesar with a new key of K1+K2.

5. Compared to original Caesar’s cipher, is there any performance drop off?

Yes, due to the implementation of the code. Since we need to call the enc and dec functions twice. Which is inefficient in our case, since single Caesar is equal in terms of effectiveness.

**Affine Cipher**

Q1, Q3: The code has been implemented on the 4th script.

Q2: Affine cipher is more secure than Caesar cipher since it uses a more complex transformation, making it harder to crack.

**Frequency analysis:**

The fifth script is a simple substitution of the most common words of English alph. characters to the ones that analyzed in the ciphertext.  
  
This method was not beneficial.  
  
So I used this steps to decrypt this cipher  
  
  
**Single character words(fillers): Q = A**

OZ’L = IT’S

O -> I Z -> T L -> S = > IT’S

G -> O after substituting the previous ones. Cause of “ZG” cipher text =>TO

After that the cipher text ZIQZ’L gave us T\*AT’S => I->H. = THAT’S

After that IGV and we get HO\* => V=W =>HOW

VIGD -> WHO\*. => w came up with the work WHOM and D=>M

VIOEI => WHI\*H => E =>C => WHICH

DXEI => M\*CH => X = U =>MUCH

…..  
  
  
Original text is:

1. What is the decrypted message from the ciphertext above?

NOBODY IS GONNA HIT AS HARD AS LIFE, BUT IT AIN'T HOW HARD YOU CAN HIT.IT’S HOWHARD YOU CAN GET HIT AND KEEP MOVING FORWARD. IT'S HOW MUCH YOU CAN TAKE, AND KEEP MOVING

FORWARD.THAT'S HOW WINNING IS DONE." IN WHICH MOVIE WAS THIS QUOTE SAID, WHO SAID IT AND

TO WHOM WAS HE SPEAKING TO? HINT: A LONG RUNNING SERIES FOR BOXING

1. What do you think are the most common letters in the above cipher? Is it different from the most common letter in the English language?

- The most common letter on the cipher text was O when we ran the frequency analysis programme. By extension of that we do believe that the plaintext might show more often some least common used letters of the English language.

3. What do you think are the least common letters in the above cipher? Is it different from the least common letter in english language?

- On this particular instance, after running a frequency analysis on the deciphered text letters like XYZ had the lowest among all the letters of the alphabet. So, this seems to be a valid assumption for our plaintext.

4. What are the answers to the questions in the decrypted message?

Rocky balboa to his son! Rocky Balboa 2006 film

5. How long did it take you to crack the cipher? Do you think this is a secure approach to encrypting messages?

Around 3 hours, after paying attention to the word patterns of the cipher text I was able to crack the cipher step by step.