# ASSIGNMENT-3

**Team Name**: codemeisters

**Members:**

HITESH KUMAR (170305)

VIJIT MALIK (170791)

PRAVEEN KUMAR (170504)

**STEPS:**

## 1. IDENTIFICATION OF THE TYPE OF ENCRYPTION.

We entered go-enter-pluck-back-give-back-back-thrnxxtzy-read in the command and came across the following cipher text.

*frcspsr dy nkao lyxs autt ff stslslx fm kns lgsecpm zlarar nslaaxqv an qks ktys. vy kcscx, qcx daxq cucy md frcspaqn vks lsztt yk qab olcn fm kns swta icdcdr. nkl szarna yd nsk ocwb scq ac ltuclm uanm kye. dqax nkb scvau ocqx kncn utat tsm nye yne yd nsk ocwls. an ueytx bpcs myc e bcvoaacq, yq tsln lkcq ciddcn! ry vy knryekv, lzspc nks czlluryx:*

*aap_niviv_dr*

First of all, we thought the problem will be solved using DES but we thought that we would require some plain text and cipher text samples to solve using this because we didn't get anything like that.

Then sir told in class that the problem is similar to the previous two assignments. Previous two assignments were based on Substitution cipher and Vigenère cipher (which is an example of substitution cipher).

Using the frequency analysis tool we determined that it has to be a monoalphabetic substitution cipher. However, as we had done substitution cipher in Assignment-1 it was logical to not go for that same cipher again. Also using the methods that we used in assignments 1 and 2 we were not getting any interpretable results. According to the sequence in which sir taught in class we went for the Substitution Permutation cipher.

## 2. BREAKING THE ENCRYPTED TEXT:

To solve problem using this, we need a key. To guess the key length we counted the total number of alphabets in the paragraph. It came out to be 270. So 270 should be multiple of key length.

But after checking the length of the password, it was 10. So 10 should be multiple of key length. So the key length can be 2,5 or 10. We tried each of them. Suppose we take one of them as n

Total permutations for key length n are n!.

Firstly we needed to generate ciphertext after interacting with each of the 120 permutations. We used the code written in generate_ciphers.c (permute function was taken from geeks for geeks referred from https://www.geeksforgeeks.org/write-a-c-program-to-print-all-permutations-of-a-given-string/) )to generate the permuted ciphers and their corresponding permutations in the ciphers.txt file.

After doing this we used the hill-climbing algorithm and simply applied it on all the 120 ciphertexts.

We used hill-climbing, algorithm something like this:

- We generated a random key of 26 alphabets from a-z and call that parent key and then deciphered the ciphertext using this key and rated the fitness of the deciphered text using this key and stored it somewhere.
- After this, we changed a little bit in this key by swapping two elements and again rated the fitness using this key.
- If the fitness score of this new key is greater than previous then discard the previous key and call this new key as parent key.
- Similarly, we can iterate multiple times and as the cycle proceeds, the key becomes better until the solution appears.

(reference from : http://practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-simple-substitution-cipher/ )

The fitness of a plain text was measured using the English quadgram metric.

(reference from: http://practicalcryptography.com/cryptanalysis/text-characterisation/quadgrams/#a-python-implementation)

The complete code is written in the file codebreakA3.py.

## 3. RESULTS:

When we tried using 2 as the key length we did not get any interpretable plain text and we were not surprised because key length 2 is just too small. So next we tried using it for 5. Finally, we got the permutation key as 12435 and the decoded text as:

“breaker of this code will be blessed by the squeaky spirit residing in the hole. go ahead, and find a way of breaking the spell on him cast by the evil jaffar. the spirit of the caveman is always with you. find the magic wand that will let you out of the caves. it would make you a magician no less than jaffar! to go through speak the password: iit_kjgjf_gr”

For this, we got the key as:

['C', 'F', 'O', 'X', 'S', 'D', 'V', 'K', 'A', 'I', 'P', 'T', 'B', 'Q', 'Y', 'Z', 'G', 'R', 'L', 'N', 'E', 'W', 'U', 'H', 'M', 'J'].