Security Lab
Lab Assignment No. 2

Aim: Write down the algorithm for breaking the Mono-alphabetic Substitution Cipher using Frequency analysis method.

Substitution ciphers are probably the most common form of cipher. They work by replacing each letter of the plaintext (and sometimes punctuation marks and spaces) with another letter (or possibly even a random symbol).

A **monoalphabetic substitution** cipher, also known as a simple substitution cipher, relies on a fixed replacement structure. That is, the substitution is fixed for each letter of the alphabet. Thus, if "a" is encrypted to "R", then every time we see the letter "a" in the plaintext, we replace it with the letter "R" in the ciphertext.

A simple example is where each letter is encrypted as the next letter in the alphabet: "a simple message" becomes "B TJNQMF NFTTBHF". In general, when performing a simple substitution manually, it is easiest to generate the ciphertext alphabet first, and encrypt by comparing this to the plaintext alphabet.

The process of **frequency analysis** uses various subtle properties of the language, and for this reason, it is near impossible to have a computer do all the work. The methodology behind frequency analysis relies on the fact that in any language, each letter has its own personality. The most obvious trait that letters have is the frequency with which they appear in a language. Clearly in English the letter "Z" appears far less frequently than, say, "A". In times gone by, if you wanted to find out the frequencies of letters within a language, you had to find a large piece of text and count each frequency.

Algorithm:

STEP 1: Ask the user to enter a plain text.
STEP 2: Generate a random number for a key to denote the required shift.
STEP 3: Call the function to encrypt the given plain text.
STEP 4: Traverse the given plain text one character at a time.
STEP 5: For each character, transform the given character as per the above required shift.
STEP 6: Return the encrypted text generated.
STEP 7: Call the function to decrypt the generated encrypted text.
STEP 8: Initialize a string say T as “ETAOINSHRDLCUMWFGYPBVKJXQZ”.
STEP 9: Find the frequency of each character and store it in a variable.
STEP 10: Iterate over the range [0,26]
STEP 11: Find the most occurring element in the encrypted text and store it in a variable.

STEP 12: Find the difference between the ch and i^{th} character of the string T and store it in a variable.
STEP 13: Iterate over the characters of encrypted text, and shift all characters by x and then push the obtained string into an array.
STEP 14: Print the strings obtained in the array.

Code:

```
import random

# Function to encrypt a plain text using Caesar Cipher
def encryption(plain_text, key):
    encrypted_text = ""

    # Traverse the plain text
    for i in range(len(plain_text)):
        char = plain_text[i]

        # Encryption of uppercase letters
        if (char.isupper()):
            encrypted_text += chr((ord(char) + key - 65) % 26 + 65)

        # Keeping the whitespace as it is
        elif (char == ' '):
            encrypted_text += ' '

        # Encryption of lowercase letters
        else:
            encrypted_text += chr((ord(char) + key - 97) % 26 + 97)

    return encrypted_text

# Function to decrypt a monoalphabetic substitution cipher using the Frequency Analysis
def printString(S, N):
            # Store final 26 possible deciphered plain text
            plain_text = [None] * 26

             # Store the frequency of each letter in encrypted text
            frequency = [0] * 26
```

```
 # Store the frequency of each letter in encrypted text in descending order
frequency_sort = [None] * 26

# Store which letter is already used
used = [0] * 26

# Traverse the encrypted text
for i in range(N):
        if (S[i] != ' '):
                frequency[ord(S[i]) - 65] += 1

# Copy the array of frequencies
for i in range(26):
        frequency_sort[i] = frequency[i]

# Stores the string formed from concatenating the english letters in the decreasing frequency in the English language
T = "ETAOINSHRDLCUMWFGYPBVKJXQZ"

# Sort the frequency array in descending order
frequency_sort.sort(reverse = True)

for i in range(26):
        ch = -1

        for j in range(26):
                if (frequency_sort[i] == frequency[j] and used[j] == 0):
                        used[j] = 1
                        ch = j
                        break

        if ch == -1:
                break

        # Store the numerical equivalent of letter at ith index of array letter_frequency
        x = ord(T[i]) - 65
```

```
            # Calculate the probable shift used in monoalphabetic cipher
            x = x - ch

            # Temporary string to generate one plaintext at a time
            curr = ""

            # Generate the probable i^th plaintext string using the shift calculated above
            for k in range(N):
                if (S[k] == ' '):
                    curr += " "
                    continue

                # Shift the kth letter of the cipher by x
                y = ord(S[k]) - 65
                y += x

                if (y < 0):
                    y += 26
                if (y > 25):
                    y -= 26

                # Add the kth shifted letter to temporary string
                curr += chr(y + 65)

            plain_text[i] = curr

        # Print the 26 generated strings
        for i in range(26):
            print(plain_text[i])

plain_text = input("Enter the plain text: ")
plain_text = plain_text.upper()

key = random.randint(1,26)

# Function call of encryption
S = encryption(plain_text, key)
N = len(S)
```

```
print("Encrypted text: ",S)

# Function call of decryption
printString(S,N)
```

Output:

```
PS D:\CNS Lab Experiments> & "C:/Users/Ninad Rao/AppData/Local/Programs/Python/Python39/python.exe" "d:/CNS Lab Experiments/assignment2.py"
Enter the plain text: My name is Ninad Rao
Encrypted text:  TF UHTL PZ UPUHK YHV
QC REQI MW RMREH VES
SE TGSK OY TOTGJ XGU
EQ FSEW AK FAFSV JSG
OA PCOG KU PKPCF TCQ
WI XKWO SC XSXKN BKY
WI XKWO SC XSXKN BKY
AM BOAS WG BWBOR FOC
FR GTFX BL GBGTW KTH
MY NAME IS NINAD RAO
XJ YLXP TD YTYLO CLZ
EQ FSEW AK FAFSV JSG
UG VIUM QA VQVIL ZIW
LX MZLD HR MHMZC QZN
CO DQCU YI DYDQT HQE
LX MZLD HR MHMZC QZN
SE TGSK OY TOTGJ XGU
RD SFRJ NX SNSFI WFT
IU JWIA EO JEJWZ NWK
WI XKWO SC XSXKN BKY
HT IVHZ DN IDIVY MVJ
AM BOAS WG BWBOR FOC
NZ OBNF JT OJOBE SBP
LX MZLD HR MHMZC QZN
YK ZMYQ UE ZUZMP DMA
NZ OBNF JT OJOBE SBP
VH WJVN RB WRWJM AJX
```

Conclusion: Thus we understood the algorithm for breaking the Mono-alphabetic Substitution Cipher using Frequency analysis method.