Experiment no. 1

Page No.
Date

Name :- Jay Parmar Roll = B-3
Sub :- CNS - Security lab

Aim :- Breaking the mono-alphabetic substitution cipher using frequency analysis method using Python.

Theory :-

To decide monoalphabetic cipher we should use Frequency analysis. In monoalphabetic cipher every character is replaced with a unique other character in the set For example in english. If E is replaced with k then every occurance of E is replaced with k. So frequency of plain text character E is same as Frequency of cipher text character k. Hence For decoding k is replaced with E.

For implementation, First take large data set of character in your language For example For english take more paragraphs From news etc and Find Frequency of each character i.e a, b, c, d, e, ... z

in source language sort it from low to high.

→ Take cipher text to be decoded Find Frequencies of each character & sort it from low to high.

→ Compare both sorted lists & pair the character in plain & cipher texts.

→ Perform replacement Almost you will get most probable plain text.

→ The disadvantage in this method if two character have almost same Frequency then mapping may be swapped.

→ If we were to just not all the letters in order & replace, them as in the Frequencies, it would likely produce jibberish. The codebreaker has to use other personality traits of the letter to decrypt the message This may include to looking at common pairs of letters there aren't many 2 letter words.

there are only a few letter which appears as doubles These are only two sensical words made fo a single

Conclusion: we successFully implemeted breaking the mono-alphabetic substitution cipher using Frequency analysis method using python

PROGRAM :

```
from operator import itemgetter

letterFrequency = [

[12.00, 'E'], [9.10, 'T'],

[8.12, 'A'], [7.68, 'O'],

[7.31, 'I'], [6.95, 'N'],

[6.28, 'S'], [6.02, 'R'],

[5.92, 'H'], [4.32, 'D'],

[3.98, 'L'], [2.88, 'U'],

[2.71, 'C'], [2.61, 'M'],

[2.30, 'F'], [2.11, 'Y'],

[2.09, 'W'], [2.03, 'G'],

[1.82, 'P'], [1.49, 'B'],

[1.11, 'V'], [0.69, 'K'],
```

[0.17, 'X'], [0.11, 'Q'],

[0.10, 'J'], [0.07, 'Z']]

plain_to_cipher = { "a": "l", "b": "f",
"c": "w", "d": "o",

"e": "a", "f": "y",

"g": "u", "h": "i",

"i": "s", "j": "v",

"k": "z", "l": "m",

"m": "n", "n": "x",

"o": "p", "p": "b",

"q": "d", "r": "c",

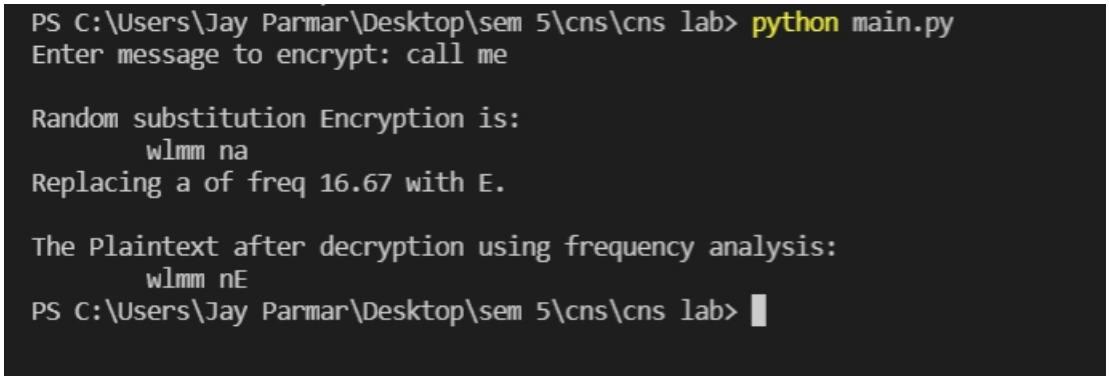
"s": "r", "t": "j",

"u": "t", "v": "q",

```
"w": "e", "x": "g",

"y": "h", "z": "k",

}

cipher_to_plain = {v: k for k, v in plain_to_cipher.items()}
alphabet = "qwertyuioplkjhgfdsazxcvbnm"

message = input("Enter message to encrypt: ")
message = message.lower()
ciphertext = ""

for c in message:

        if c not in alphabet:

                ciphertext += c
        else:
                ciphertext += plain_to_cipher[c]

print("\nRandom substitution Encryption is: \n\t{}".format(ciphertext))
letter_list = []
cipher_len = 0

for c in ciphertext:
```

```
        if c in alphabet:

                cipher_len += 1

if c not in letter_list:
        letter_list.append(c)



letter_freq = []
for c in letter_list:
        letter_freq.append([round(ciphertext.count(c) / cipher_len * 100, 2), c])

letter_freq = sorted(letter_freq, key=itemgetter(0), reverse=True)
decrypted_plaintext = ciphertext

index = 0

for f, c in letter_freq:

        print("Replacing {} of freq {} with {}.".format(c, f,
letterFrequency[index][1]))
        decrypted_plaintext = decrypted_plaintext.replace(c,
letterFrequency[index][1])
        index += 1
        print("\nThe Plaintext after decryption using frequency
analysis:\n\t{}".format(decrypted_plaintext))
```

OUTPUT :



CONCLUSION :

Hence we learned and implemented breaking monoalphabetic substitution cipher using frequency analysis method in python.