**IMPLEMENTING SIMPLE CAESAR CIPHERS AND BREAKING IT USING FREQUENCY ANALYSIS**

**SUBJECT NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

**SUBJECT CODE: CS6008**

**MODULE: 5**

|  |  |
| --- | --- |
| **NAME** | BHUVANESHWAR S |
| **REG.NO** | 2019103513 |
| **DATE** | 05/04/2022 |

**AIM :**

To implement a simple Caesar cipher and crack the cipher text using frequency analysis.

**TOOL INVOLVED:**

* Python
* Terminal
* Visual Studio Code

**PROBLEM DESCRIPTION:**

Caesar cipher is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. It is also known as additive cipher.

The encryption can also be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A → 0, B → 1, ..., Z → 25. Encryption of a letter x by a shift n can be described mathematically as



For decryption



Caesar cipher can be easily broken using

* brute-force attack
* frequency analysis

Brute-force attack:

A brute-force attack tries every possible decryption key for a cipher. Nothing stops a cryptanalyst from guessing one key, decrypting the ciphertext with that key, looking at the output, and then moving on to the next key if they didn’t find the secret message.

Frequency analysis:

Frequency analysis is one of the known ciphertext attacks. It is based on the study of the frequency of letters or groups of letters in a ciphertext. The attacker usually checks some possibilities and makes some substitutions of letters in ciphertext. He looks for possible appearing words and based on that makes more substitutions. Using computers, it is possible to try a lot of combinations in relative short time.

KEY = ((index of most frequency letter) – (index of character ‘e’ i.e. 4))mod 26

**INPUT:**

Getting an input cipher text from the user in terminal.

**OUTPUT:**

Crack the cipher text using frequency analysis.

**SCREENSHOT:**

Encryption.py

alphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

*def* encryption(*plain\_text*,*key*):

    cipher\_text = ''

    for i in range(len(*plain\_text*)):

        char = *plain\_text*[i]

        # print(char)

        if char.isupper():

            index = alphabet.index(char.upper())

            cipher\_text += alphabet[(index+*key*)%26].upper()

        elif char.islower():

            index = alphabet.index(char.upper())

            cipher\_text += alphabet[(index+*key*)%26].lower()

        else:

            cipher\_text += char

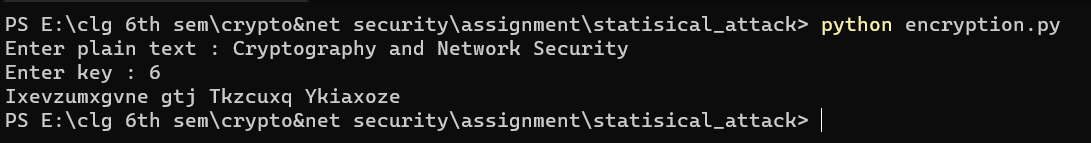
    return cipher\_text

plain\_text = input("Enter plain text : ")

key = int(input("Enter key : "))

print(encryption(plain\_text,key))

OUTPUT



Here, the plain text is ‘Cryptography and Network security’ to encrypt the message using key value 6.

Decryption.py

LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

*def* decryption(*cipher\_text*,*key*):

    plain\_text = ''

    for i in range(len(*cipher\_text*)):

        char = *cipher\_text*[i]

        if char.isupper():

            index = LETTERS.index(char.upper())

            plain\_text += LETTERS[(index-*key*)%26].upper()

        elif char.islower():

            index = LETTERS.index(char.upper())

            plain\_text += LETTERS[(index-*key*)%26].lower()

        else:

            plain\_text += char

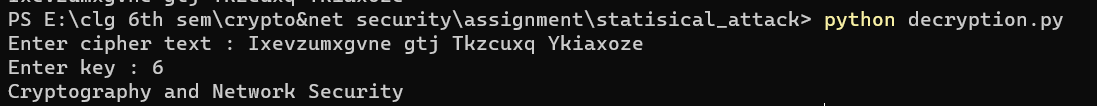
    return plain\_text

cipher\_text = input("Enter cipher text : ")

key = int(input("Enter key : "))

print(decryption(cipher\_text,key))

OUTPUT



Decrypting the text using the key value 6

attack.py

from audioop import reverse

LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

*def* frequency\_analysis(*cipher\_text*):

*cipher\_text* = *cipher\_text*.upper()

    letter\_frequency = {}

    for i in LETTERS:

        letter\_frequency[i] = 0;

    for i in *cipher\_text*:

        if i in LETTERS:

            letter\_frequency[i] += 1

    return letter\_frequency

*def* decryption(*cipher\_text*,*key*):

    plain\_text = ''

    for i in range(len(*cipher\_text*)):

        char = *cipher\_text*[i]

        if char.isupper():

            index = LETTERS.index(char.upper())

            plain\_text += LETTERS[(index-*key*)%26].upper()

        elif char.islower():

            index = LETTERS.index(char.upper())

            plain\_text += LETTERS[(index-*key*)%26].lower()

        else:

            plain\_text += char

    return plain\_text

*def* caeser\_crack(*cipher\_text*):

    letter\_frequency = frequency\_analysis(*cipher\_text*)

    print(letter\_frequency)

    letter\_frequency = {k: v for k, v in sorted(letter\_frequency.items(), *key*=*lambda* *item*: *item*[1],*reverse*=True)}

    for x,y in letter\_frequency.items():

        if y != 0:

            index\_l = LETTERS.index(x)

            index\_e = LETTERS.index('E')

            key = (index\_l - index\_e)%26

            print("\n[+]\tMOST FRQUENCE LETTER : " ,x,"\tKEY : ",key)

            print(decryption(*cipher\_text*,key))

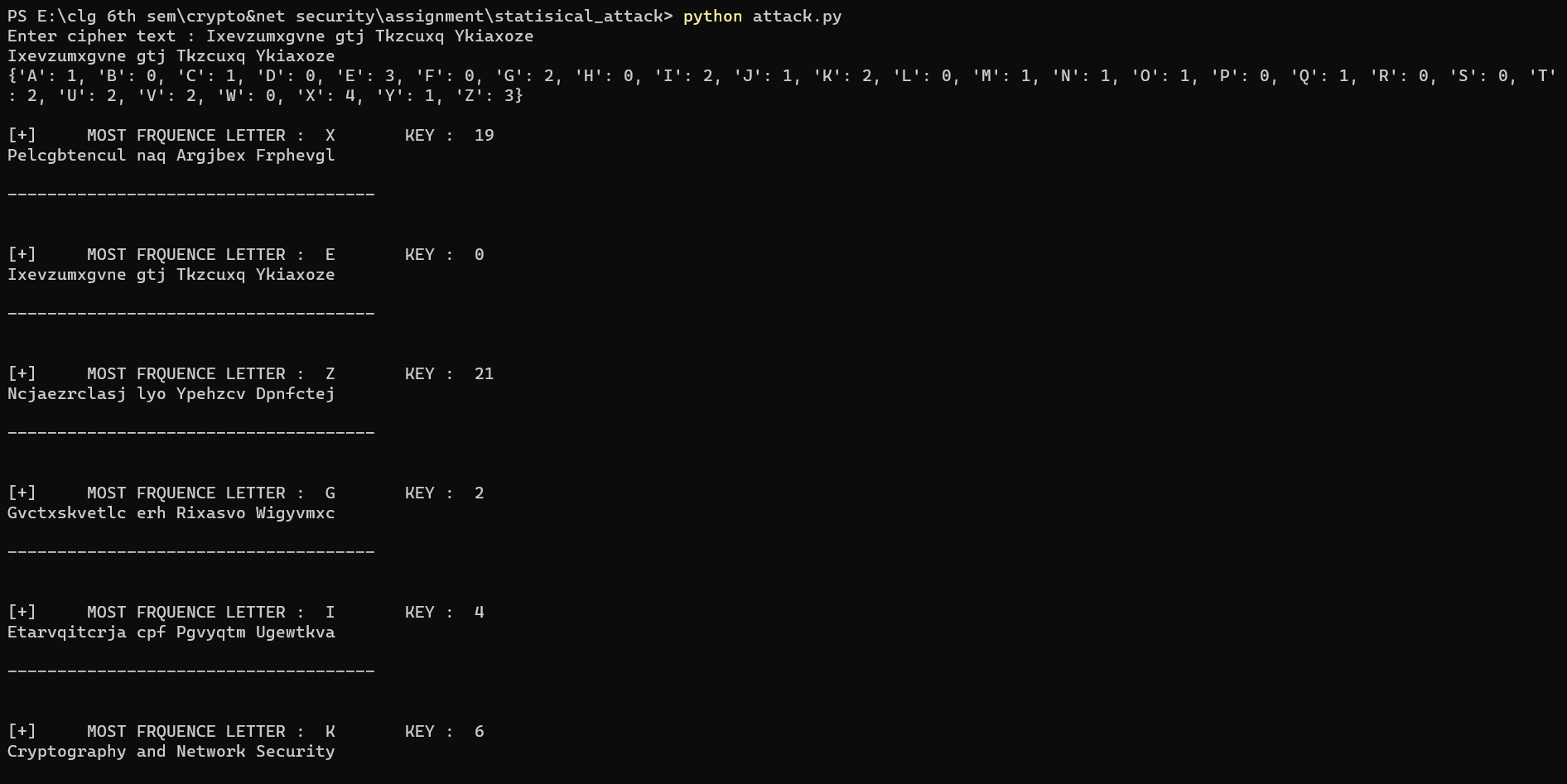
            print("\n-------------------------------------\n")

cipher\_text = input("Enter cipher text : ")

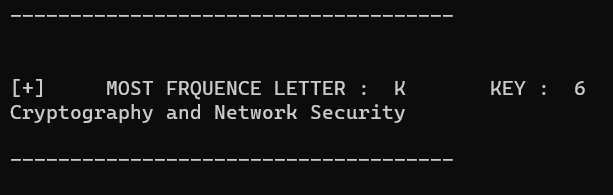
print(cipher\_text)

caeser\_crack(cipher\_text)

OUTPUT



It display the number of frequency for each letter in cipher text.



Hence we crack the plain text using frequency analysis and found the value of key.