# **Name: Abdurrahman Qureshi**

# **Roll No: 242466**

Practical No: 1

Date Of Performance: 07/07/2025

Aim: To encrypt and decrypt messages using different mono-alphabetic cipher techniques and frequency analysis

Ceaser Cipher:

CODE:

alphabets = "ABCDEFGHIJKLMNOPQRSTUVWXYZ "

def ceaser\_cipher\_encrypt(plain\_text: str, key):

    if len(plain\_text) == 0: return

    plain\_text = plain\_text.upper()

    plain\_text = plain\_text.strip()

    encrypted\_message = ""

    characters\_in\_plain\_text = []

    for i in plain\_text:

        mapped\_index = alphabets.index(i)

        encrypted\_message = encrypted\_message + alphabets[(mapped\_index + key) % 27]

        if i not in characters\_in\_plain\_text:

            characters\_in\_plain\_text.append(i)

    print("Length of the plain text:", len(plain\_text))

    for i in characters\_in\_plain\_text:

        if i == " ":

            print(f"Frequency of ' ' in plain text is {(len(plain\_text) / plain\_text.count(i))}%")

        else:

            print(f"Frequency of {i} in plain text is {(len(plain\_text) / plain\_text.count(i))}%")

    print(characters\_in\_plain\_text)

    return encrypted\_message

def ceaser\_cipher\_decrypt(cipher\_text: str, key):

    if len(cipher\_text) == 0: return

    decrypted\_message = ""

    cipher\_text = cipher\_text.upper()

    cipher\_text = cipher\_text.strip()

    for i in cipher\_text:

        mapped\_index = alphabets.index(i)

        decrypted\_message = decrypted\_message + alphabets[(mapped\_index - key) % 27]

    return decrypted\_message

original\_message = "Death is an old friend"

print("Original Message: ", original\_message)

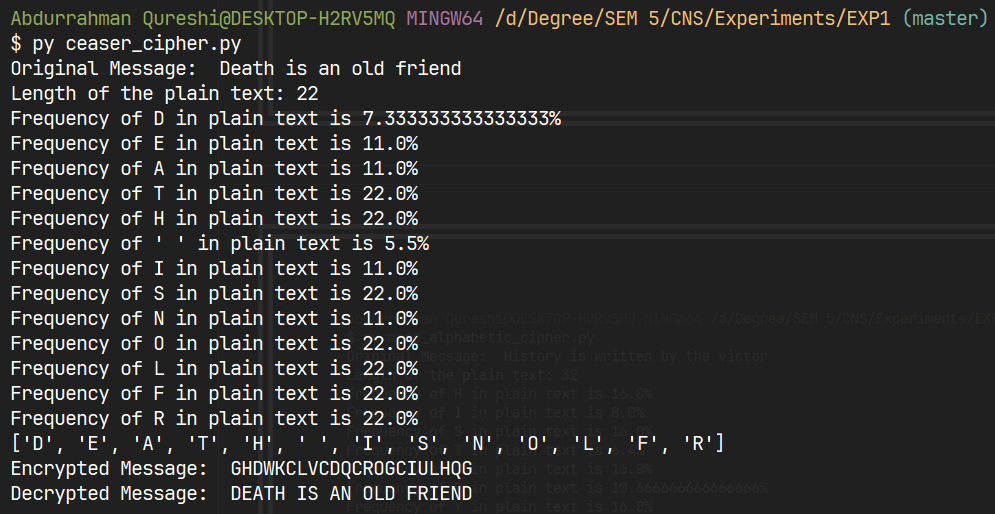
encrypted\_message = ceaser\_cipher\_encrypt(original\_message, 3)

print("Encrypted Message: ", encrypted\_message)

decrypted\_message = ceaser\_cipher\_decrypt(encrypted\_message, 3)

print("Decrypted Message: ", decrypted\_message)

OUTPUT:



Mono-Alphabetic Cipher:

CODE:

alphabets = "ABCDEFGHIJKLMNOPQRSTUVWXYZ "

substituted\_alphabets = "PJBTDGQRVHXCKLZOMFYWNAIUSE/"

def mono\_alphabetic\_encrypt(plain\_text: str):

    if len(plain\_text) == 0: return

    plain\_text = plain\_text.upper()

    plain\_text = plain\_text.strip()

    encrypted\_text = ""

    characters\_in\_plain\_text = []

    for i in plain\_text:

        mapped\_index = alphabets.index(i)

        encrypted\_text = encrypted\_text + substituted\_alphabets[mapped\_index]

        if i not in characters\_in\_plain\_text:

            characters\_in\_plain\_text.append(i)

    print("Length of the plain text:", len(plain\_text))

    for i in characters\_in\_plain\_text:

        if i == " ":

            print(f"Frequency of ' ' in plain text is {(len(plain\_text) / plain\_text.count(i))}%")

        else:

            print(f"Frequency of {i} in plain text is {(len(plain\_text) / plain\_text.count(i))}%")

    print(characters\_in\_plain\_text)

    return encrypted\_text

def mono\_alphabetic\_decrypt(cipher\_text: str):

    if len(cipher\_text) == 0: return

    cipher\_text = cipher\_text.upper()

    cipher\_text = cipher\_text.strip()

    decrypted\_text = ""

    for i in cipher\_text:

        mapped\_index = substituted\_alphabets.index(i)

        decrypted\_text = decrypted\_text + alphabets[mapped\_index]

    return decrypted\_text

original\_message = "History is written by the victor"

print("Original Message: ", original\_message)

encrypted\_message = mono\_alphabetic\_encrypt(original\_message)

print("Encrypted Message: ", encrypted\_message)

decrypted\_message = mono\_alphabetic\_decrypt(encrypted\_message)

print("Decrypted Message: ", decrypted\_message)

OUTPUT:

