Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 1

Date Of Performance: 07/07/2025

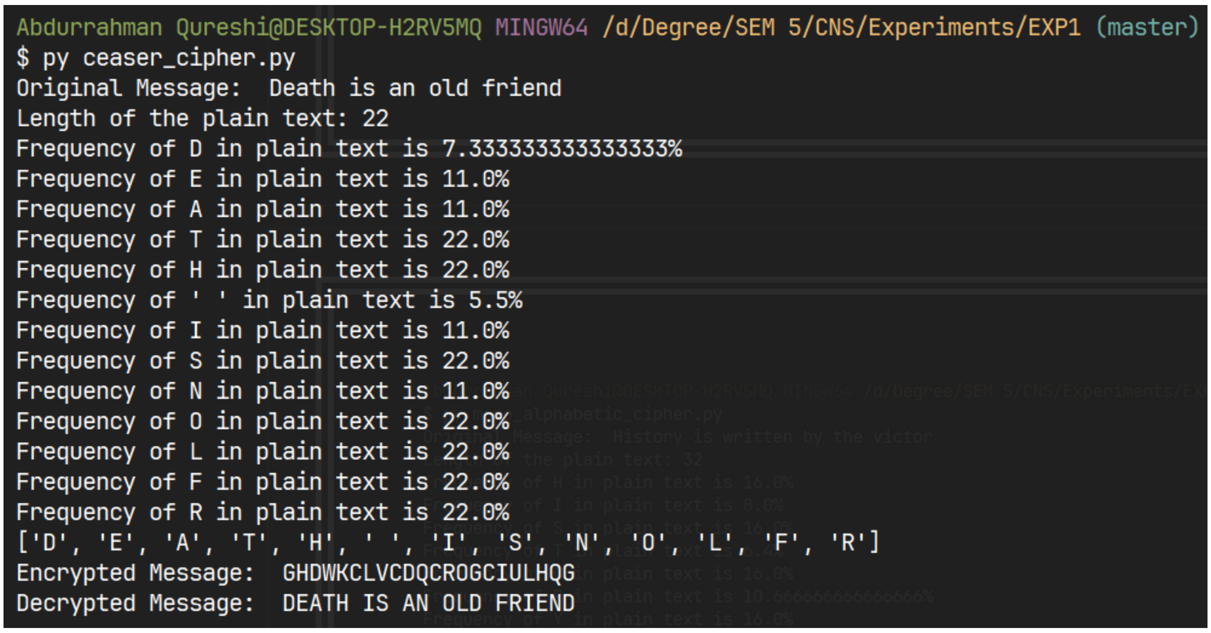
Aim: To encrypt and decrypt messages using different mono-alphabetic cipher techniques and frequency analysis

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:



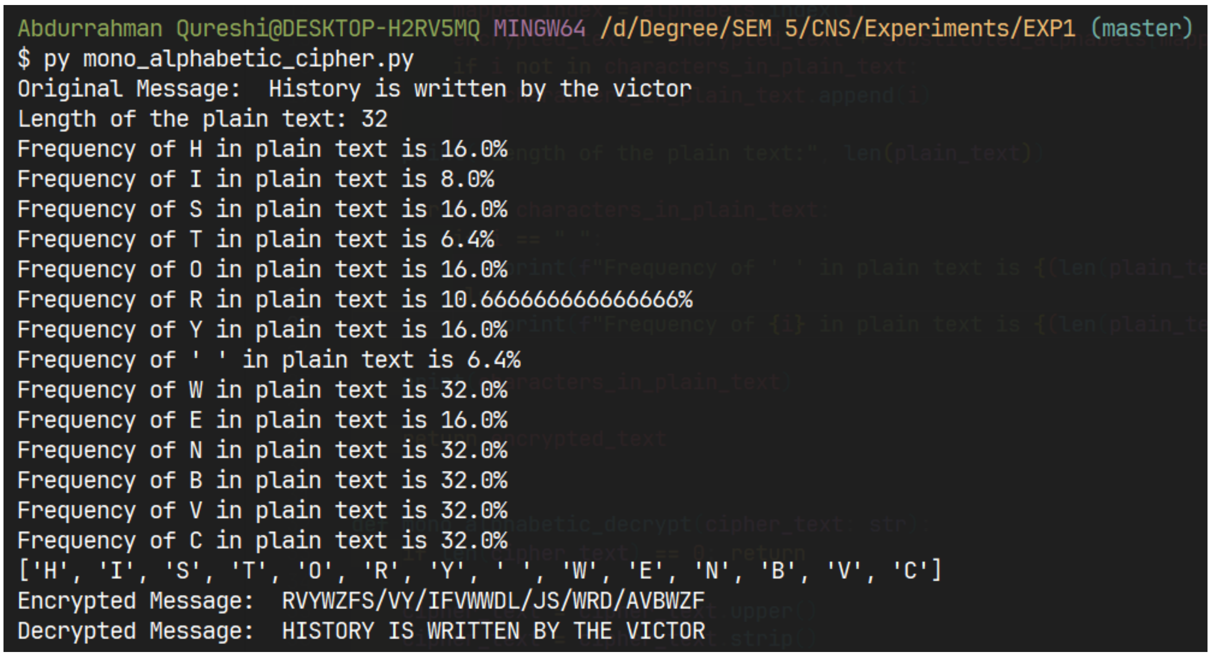
```
Abdurrahman Qureshi@DESKTOP-H2RV5MQ MINGW64 /d/Degree/SEM 5/CNS/Experiments/EXP1 (master)
$ py ceaser_cipher.py
Original Message:  Death is an old friend
Length of the plain text: 22
Frequency of D in plain text is 7.333333333333333%
Frequency of E in plain text is 11.0%
Frequency of A in plain text is 11.0%
Frequency of T in plain text is 22.0%
Frequency of H in plain text is 22.0%
Frequency of ' ' in plain text is 5.5%
Frequency of I in plain text is 11.0%
Frequency of S in plain text is 22.0%
Frequency of N in plain text is 11.0%
Frequency of O in plain text is 22.0%
Frequency of L in plain text is 22.0%
Frequency of F in plain text is 22.0%
Frequency of R in plain text is 22.0%
['D', 'E', 'A', 'T', 'H', ' ', 'I', 'S', 'N', 'O', 'L', 'F', 'R']
Encrypted Message:  GHDWKCLVCDQCROGCIULHQG
Decrypted Message:  DEATH IS AN OLD FRIEND
```

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:



```
Abdurrahman Qureshi@DESKTOP-H2RV5MQ MINGW64 /d/Degree/SEM 5/CNS/Experiments/EXP1 (master)
$ py mono_alphabetic_cipher.py
Original Message:  History is written by the victor
Length of the plain text: 32
Frequency of H in plain text is 16.0%
Frequency of I in plain text is 8.0%
Frequency of S in plain text is 16.0%
Frequency of T in plain text is 6.4%
Frequency of O in plain text is 16.0%
Frequency of R in plain text is 10.666666666666666%
Frequency of Y in plain text is 16.0%
Frequency of ' ' in plain text is 6.4%
Frequency of W in plain text is 32.0%
Frequency of E in plain text is 16.0%
Frequency of N in plain text is 32.0%
Frequency of B in plain text is 32.0%
Frequency of V in plain text is 32.0%
Frequency of C in plain text is 32.0%
['H', 'I', 'S', 'T', 'O', 'R', 'Y', ' ', 'W', 'E', 'N', 'B', 'V', 'C']
Encrypted Message:  RVYWZFS/VY/IFVWWDL/JS/WRD/AVBWZF
Decrypted Message:  HISTORY IS WRITTEN BY THE VICTOR
```