1. Assume you intercepted the following ciphertext. Using a statistical attack, find the plaintext

"XLILSYWIMWRSAJSVWEPIJSVJSYVQMPPMSRHSPPEVWMXMWASVX-LQSVILYVVCFIJSVIXLIWIPPIVVIGIMZIWQSVISJJIVW"

Sol:

```
kali@kali: ~
File  Actions  Edit  View  Help
from collections import Counter
def frequency_analysis(ciphertext):
    frequency = Counter(ciphertext)
    return frequency

def decrypt_with_substitution(ciphertext, substitutions):
    plaintext = ''.join(substitutions.get(char, char) for char in ciphertext)
    return plaintext
ciphertext = "XLILSYWIMWRSAJSVWEPIJSVJSYVQMPPMSRHSPPEVWMXMWASVX-LQSVILYVVCFIJSVIXLIWIPPIVVIGIMZIWQSVISJJIVW"
freq = frequency_analysis(ciphertext)
print("Letter Frequencies:", freq)
substitutions = {
    'X': 'E',
    'L': 'T',
    'I': 'A',
    'S': 'O',
    'Y': 'H',
    'W': 'N',
    'M': 'R',
    'P': 'I',
    'R': 'S',
    'J': 'D',
    'V': 'U',
    'Q': 'G',
    'A': 'Y',
}
plaintext = decrypt_with_substitution(ciphertext, substitutions)
print("Decrypted Plaintext:", plaintext)
```

Output:

```
┌──(kali㉿kali)-[~]
└─$ vi lab6.py

┌──(kali㉿kali)-[~]
└─$ python lab6.py
Letter Frequencies: Counter({'I': 14, 'V': 13, 'S': 12, 'W': 8, 'P': 7, 'M': 6, 'J': 6,
 'L': 5, 'X': 4, 'Y': 3, 'Q': 3, 'R': 2, 'A': 2, 'E': 2, 'H': 1, '-': 1, 'C': 1, 'F': 1
, 'G': 1, 'Z': 1})
Decrypted Plaintext: ETATOHNARNSOYDOUNEIADOUDOHUGRIIROSHOIIEUNRERNYOUE-TGOUATHUUCFADOUA
ETANAIIAUUAGARZANGOUAODDAUN
```

2. Write a Python script to encrypt using Rail Fence (Zig zag) with three rows and with key (ONE).

Sol:

```
kali@kali: ~
File  Actions  Edit  View  Help
def rail_fence_encrypt(message, key):
    rail = [['\n' for _ in range(len(message))]
            for _ in range(key)]
    row, col, direction = 0, 0, 1
    for char in message:
        rail[row][col] = char
        col += 1
        if row == 0:
            direction = 1
        elif row == key - 1:
            direction = -1
        row += direction
    encrypted_message = ''.join([''.join(r) for r in rail]).replace('\n', '')
    return encrypted_message
key = 3
message = "HELLOTHERE"
encrypted = rail_fence_encrypt(message, key)
print("Encrypted Message:", encrypted)
```

Output:

```
┌──(kali㉿kali)-[~]
└─$ vi lab6.py

┌──(kali㉿kali)-[~]
└─$ python lab6.py
Encrypted Message: HORELTEELH
```

3. Write a python script to encrypt columnar transposition

Sol:

```
kali@kali: ~
File  Actions  Edit  View  Help
def columnar_transposition_encrypt(message, key):
    message = message.replace(" ", "")
    key_length = len(key)
    columns = ['' for _ in range(key_length)]
    for i in range(len(message)):
        columns[i % key_length] += message[i]
    sorted_indices = sorted(range(len(key)), key=lambda k: key[k])
    encrypted_message = ''.join(columns[i] for i in sorted_indices)
    return encrypted_message
key = "KEY"
message = "HELLO WORLD"
encrypted = columnar_transposition_encrypt(message, key)
print("Encrypted Message:", encrypted)
```

Output:

```
┌──(kali㉿kali)-[~]
└─$ vi lab6.py

┌──(kali㉿kali)-[~]
└─$ python lab6.py
Encrypted Message: EORHLODLWL
```

4. Write a Python script to decrypt Rail Fence Cipher

Sol:

```
1  2  3  4
kali@kali: ~
File  Actions  Edit  View  Help
from collections import Counter
def frequency_analysis(ciphertext):
    frequency = Counter(ciphertext)
    return frequency

def decrypt_with_substitution(ciphertext, substitutions):
    plaintext = ''.join(substitutions.get(char, char) for char in ciphertext)
    return plaintext
ciphertext = "XLILSYWIMWRSAJSVWEPIJSVJSYVQMPPMSRHSPPEVWMXMWASVX-LQSVILYVVCFIJSVIXLIWIPPIVVIGIMZIWQSVISJJIVW"
freq = frequency_analysis(ciphertext)
print("Letter Frequencies:", freq)
substitutions = {
    'X': 'E',
    'L': 'T',
    'I': 'A',
    'S': 'O',
    'Y': 'H',
    'W': 'N',
    'M': 'R',
    'P': 'I',
    'R': 'S',
    'J': 'D',
    'V': 'U',
    'Q': 'G',
    'A': 'Y',
}
plaintext = decrypt_with_substitution(ciphertext, substitutions)
print("Decrypted Plaintext:", plaintext)
```

Output:

```
┌──(kali㉿kali)-[~]
└─$ vi lab6.py

┌──(kali㉿kali)-[~]
└─$ python lab6.py
Letter Frequencies: Counter({'I': 14, 'V': 13, 'S': 12, 'W': 8, 'P': 7, 'M': 6, 'J': 6,
 'L': 5, 'X': 4, 'Y': 3, 'Q': 3, 'R': 2, 'A': 2, 'E': 2, 'H': 1, '-': 1, 'C': 1, 'F': 1
, 'G': 1, 'Z': 1})
Decrypted Plaintext: ETATOHNARNSOYDOUNEIADOUDOHUGRIIROSHOIIEUNRERNYOUE-TGOUATHUUCFADOUA
ETANAIIAUUAGARZANGOUAODDAUN
```