**1. Write a Python script to encrypt columnar transposition using keyword.**

def columnar\_transposition\_encrypt(message, keyword):

message = ''.join(message.split()).upper()

keyword = ''.join(keyword.split()).upper()

column\_order = sorted(range(len(keyword)), key=lambda k: keyword[k])

num\_rows = -(-len(message) // len(keyword))

message += 'X' \* (num\_rows \* len(keyword) - len(message))

grid = [[''] \* len(keyword) for \_ in range(num\_rows)]

index = 0

for row in range(num\_rows):

for col in range(len(keyword)):

grid[row][col] = message[index]

index += 1

ciphertext = ''

for col in column\_order:

ciphertext += ''.join(grid[row][col] for row in range(num\_rows))

message = input("Enter the message to encrypt: ")

keyword = input("Enter the keyword for encryption: ")

encrypted\_message = columnar\_transposition\_encrypt(message, keyword)

print(f"Encrypted message: {encrypted\_message}")

**2. Write a Python script to encrypt double columnar transposition.**

def columnar\_transposition(message, keyword):

message = ''.join(message.split()).upper()

keyword = ''.join(keyword.split()).upper()

column\_order = sorted(range(len(keyword)), key=lambda k: keyword[k])

num\_rows = -(-len(message) // len(keyword))

message += 'X' \* (num\_rows \* len(keyword) - len(message))

grid = [[''] \* len(keyword) for \_ in range(num\_rows)]

index = 0

for row in range(num\_rows):

for col in range(len(keyword)):

grid[row][col] = message[index]

index += 1

ciphertext = ''

for col in column\_order:

ciphertext += ''.join(grid[row][col] for row in range(num\_rows))

return ciphertext

def double\_columnar\_transposition\_encrypt(message, keyword1, keyword2):

first\_encryption = columnar\_transposition(message, keyword1)

second\_encryption = columnar\_transposition(first\_encryption, keyword2)

return second\_encryption

message = input("Enter the message to encrypt: ")

keyword1 = input("Enter the first keyword for encryption: ")

keyword2 = input("Enter the second keyword for encryption: ")

encrypted\_message = double\_columnar\_transposition\_encrypt(message, keyword1, keyword2)

print(f"Encrypted message: {encrypted\_message}")

**3. Write a Python script to encrypt the message “She is listening” using the 6-character keyword “PASCAL” with Vigenere cipher.**

def vigenere\_encrypt(plaintext, keyword):

ciphertext = ""

keyword = keyword.upper()

keyword\_length = len(keyword)

keyword\_as\_int = [ord(i) - ord('A') for i in keyword]

for i, char in enumerate(plaintext.upper()):

if char.isalpha():

char\_num = ord(char) - ord('A')

keyword\_num = keyword\_as\_int[i % keyword\_length]

encrypted\_num = (char\_num + keyword\_num) % 26

encrypted\_char = chr(encrypted\_num + ord('A'))

ciphertext += encrypted\_char

else:

ciphertext += char

return ciphertext

message = "She is listening"

keyword = "PASCAL"

encrypted\_message = vigenere\_encrypt(message, keyword)

print(f"Original message: {message}")

print(f"Keyword: {keyword}")

print(f"Encrypted message: {encrypted\_message}")

**4. Write a Python script to encrypt and decrypt Hill cipher**

def vigenere\_encrypt(plaintext, keyword):

ciphertext = ""

keyword = keyword.upper()

keyword\_length = len(keyword)

keyword\_as\_int = [ord(i) - ord('A') for i in keyword]

for i, char in enumerate(plaintext.upper()):

if char.isalpha():

char\_num = ord(char) - ord('A')

keyword\_num = keyword\_as\_int[i % keyword\_length]

encrypted\_num = (char\_num + keyword\_num) % 26

encrypted\_char = chr(encrypted\_num + ord('A'))

ciphertext += encrypted\_char

else:

ciphertext += char

return ciphertext

message = "She is listening"

keyword = "PASCAL"

encrypted\_message = vigenere\_encrypt(message, keyword)

print(f"Original message: {message}")

print(f"Keyword: {keyword}")

print(f"Encrypted message: {encrypted\_message}")