Import numpy as np

# Hill Cipher

Def hill\_cipher\_encrypt(plain\_text, key\_matrix):

Plain\_text = plain\_text.upper().replace(“ “, “”)

N = len(key\_matrix)

Plain\_text = [ord(char) – 65 for char in plain\_text]

Cipher\_text = “”

For I in range(0, len(plain\_text), n):

Block = np.array(plain\_text[i:i+n])

Encrypted\_block = np.dot(key\_matrix, block) % 26

Cipher\_text += ‘’.join([chr(char + 65) for char in encrypted\_block])

Return cipher\_text

# Polyaphabetic Cipher (Vigenère Cipher)

Def vigenere\_cipher\_encrypt(plain\_text, keyword):

Plain\_text = plain\_text.upper().replace(“ “, “”)

Keyword = keyword.upper()

Cipher\_text = “”

For I in range(len(plain\_text)):

Shift = ord(keyword[I % len(keyword)]) – 65

Cipher\_text += chr(((ord(plain\_text[i]) – 65 + shift) % 26) + 65)

Return cipher\_text

# Playfair Cipher

Def generate\_playfair\_matrix(key):

Key = key.upper().replace(“ “, “”)

Key += “ABCDEFGHIKLMNOPQRSTUVWXYZ”

Matrix = []

For char in key:

If char not in matrix:

Matrix.append(char)

Matrix = np.array(matrix).reshape(5, 5)

Return matrix

Def playfair\_cipher\_encrypt(plain\_text, key):

Plain\_text = plain\_text.upper().replace(“ “, “”)

Key\_matrix = generate\_playfair\_matrix(key)

Cipher\_text = “”

For I in range(0, len(plain\_text), 2):

Char1 = plain\_text[i]

Char2 = plain\_text[i+1] if i+1 < len(plain\_text) else ‘X’

Row1, col1 = np.where(key\_matrix == char1)

Row2, col2 = np.where(key\_matrix == char2)

If row1 == row2:

Cipher\_text += key\_matrix[row1, (col1+1)%5][0] + key\_matrix[row2, (col2+1)%5][0]

Elif col1 == col2:

Cipher\_text += key\_matrix[(row1+1)%5, col1][0] + key\_matrix[(row2+1)%5, col2][0]

Else:

Cipher\_text += key\_matrix[row1, col2][0] + key\_matrix[row2, col1][0]

Return cipher\_text

# RC5

Def rc5\_encrypt(plain\_text, key, rounds=12, w=32, r=16):

Mod = 2 \*\* w

B = len(key)

Key = [ord(char) for char in key]

L = [0] \* (b // 4)

C = [0] \* (2 \* r)

S = [0] \* (2 \* r)

For I in range(b // 4):

L[i] = (key[4\*i] << 24) + (key[4\*i+1] << 16) + (key[4\*i+2] << 8) + key[4\*i+3]

C = max(1, (max(ord(char) for char in plain\_text) + 1) // w)

Plain\_text += ‘\x00’ \* (c \* w – len(plain\_text))

A = [0] \* c

For I in range©:

A[i] = sum(ord(plain\_text[w\*i+j]) << (8 \* j) for j in range(w))

For I in range(2 \* r):

S[i] = (S[i-1] + 0xB7E15163) % mod

For I in range(3 \* max(c, 2 \* r)):

A\_i = A[I % c]

B\_i = S[2\*I % (2 \* r)]

X = (S[2\*I % (2 \* r)] + S[2\*i+1 % (2 \* r)]) % mod

X = (x << 3) | (x >> (w – 3))

T = (x \* (x + 1)) // 2

T = (t << 3) | (t >> (w – 3))

U = (A\_i ^ B\_i) + t

U = (u << 3) | (u >> (w – 3))

A[I % c] = ((A\_i ^ u) << (B\_i & (w – 1))) | ((A\_i ^ u) >> (w – (B\_i & (w – 1))))

Return ‘’.join(f”{x:08X}” for x in A)

# Example usage

Key\_matrix = np.array([[6, 24, 1], [13, 16, 10], [20, 17, 15]])

Plain\_text = “HELLO WORLD”

Hill\_cipher\_text = hill\_cipher\_encrypt(plain\_text, key\_matrix)

Print(“Hill Cipher:”, hill\_cipher\_text)

Keyword = “KEY”

Vigenere\_cipher\_text = vigenere\_cipher\_encrypt(plain\_text, keyword)

Print(“Vigenère Cipher:”, vigenere\_cipher\_text)

Key = “PLAYFAIREXAMPLE”

Playfair\_cipher\_text = playfair\_cipher\_encrypt(plain\_text, key)

Print(“Playfair Cipher:”, playfair\_cipher\_text)

Rc5\_key = “SECRETKEY”

Rc5\_text = rc5\_encrypt(plain\_text, rc5\_key)

Print(“RC5 Cipher:”, rc5\_text)