3.Play fair algorithm

Program:

def generate\_playfair\_matrix(key):

key = key.replace("J", "I") # Treat 'J' as 'I' in the key

key = key.upper()

key = "".join(dict.fromkeys(key)) # Remove duplicates while maintaining order

alphabet = "ABCDEFGHIKLMNOPQRSTUVWXYZ"

matrix = [[0] \* 5 for \_ in range(5)]

key\_index = 0

for i in range(5):

for j in range(5):

if key\_index < len(key):

matrix[i][j] = key[key\_index]

key\_index += 1

else:

for letter in alphabet:

if letter not in key and letter not in [matrix[x][y] for x in range(5) for y in range(5)]:

matrix[i][j] = letter

break

return matrix

def find\_positions(matrix, char):

for i in range(5):

for j in range(5):

if matrix[i][j] == char:

return i, j

def playfair\_encrypt(plain\_text, key):

matrix = generate\_playfair\_matrix(key)

encrypted\_text = ""

plain\_text = plain\_text.upper().replace("J", "I")

pairs = [plain\_text[i:i + 2] for i in range(0, len(plain\_text), 2)]

for pair in pairs:

if len(pair) == 1:

pair += "X"

row1, col1 = find\_positions(matrix, pair[0])

row2, col2 = find\_positions(matrix, pair[1])

if row1 == row2: # Same row

encrypted\_text += matrix[row1][(col1 + 1) % 5] + matrix[row2][(col2 + 1) % 5]

elif col1 == col2: # Same column

encrypted\_text += matrix[(row1 + 1) % 5][col1] + matrix[(row2 + 1) % 5][col2]

else: # Different row and column

encrypted\_text += matrix[row1][col2] + matrix[row2][col1]

return encrypted\_text

# Example usage

plaintext = input("Enter the text:")

key = input("Enter the key:")

encrypted\_text = playfair\_encrypt(plaintext, key)

print("Encrypted:", encrypted\_text)

Output:

