# **Name: Abdurrahman Qureshi**

# **Roll No: 242466**

Practical No: 2

Date Of Performance: 14/07/2025

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

Playfair Cipher:

CODE:

alphabets: str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

row\_1: list = []

row\_2: list = []

row\_3: list = []

row\_4: list = []

row\_5: list = []

matrix = [row\_1, row\_2, row\_3, row\_4, row\_5]

encrypted\_pairs:list = []

decrypted\_pairs:list = []

def playfair\_matrix(key: str):

    for i in key:

        if len(row\_1) != 5:

            if i == "I" or i == "J": row\_1.append("I/J")

            else: row\_1.append(i)

        elif len(row\_2) != 5:

            if i == "I" or i == "J": row\_2.append("I/J")

            else: row\_2.append(i)

        elif len(row\_3) != 5:

            if i == "I" or i == "J": row\_3.append("I/J")

            else: row\_3.append(i)

        elif len(row\_4) != 5:

            if i == "I" or i == "J": row\_4.append("I/J")

            else: row\_4.append(i)

        elif len(row\_5) != 5:

            if i == "I" or i == "J": row\_5.append("I/J")

            else: row\_5.append(i)

    for i in range(65, 91):

        char = chr(i)

        if char in key: continue

        elif char in row\_1: continue

        elif char in row\_2: continue

        elif char in row\_3: continue

        elif char in row\_4: continue

        elif char in row\_5: continue

        if char == "I" or char == "J":

            if ("I" in key or "J" in key or "I/J" in key):continue

            elif ("I" in row\_1 or "J" in row\_1 or "I/J" in row\_1):continue

            elif ("I" in row\_2 or "J" in row\_2 or "I/J" in row\_2):continue

            elif ("I" in row\_3 or "J" in row\_3 or "I/J" in row\_3):continue

            elif ("I" in row\_4 or "J" in row\_4 or "I/J" in row\_4):continue

            elif ("I" in row\_5 or "J" in row\_5 or "I/J" in row\_5):continue

            else:char = "I/J"

        if len(row\_1) < 5: row\_1.append(char)

        elif len(row\_2) < 5: row\_2.append(char)

        elif len(row\_3) < 5: row\_3.append(char)

        elif len(row\_4) < 5: row\_4.append(char)

        elif len(row\_5) < 5: row\_5.append(char)

    print()

    print("Playfair Matrix:")

    print()

    print(row\_1)

    print(row\_2)

    print(row\_3)

    print(row\_4)

    print(row\_5)

    print()

def get\_text\_pairs(text\_pairs: str, type\_pairs: list):

    i = 0

    while(i < len(text\_pairs)):

        prev\_char = text\_pairs[i - 1]

        current\_char = text\_pairs[i]

        next\_char = text\_pairs[i + 1]

        if current\_char == "X" and i == len(text\_pairs - 1):

            type\_pairs.append(text\_pairs[i] + "Z")

            break

        if i == len(text\_pairs) - 1:

            type\_pairs.append(text\_pairs[i] + "X")

            break

        if current\_char == next\_char:

            type\_pairs.append(text\_pairs[i] + "X")

            i = i + 1

        elif current\_char == prev\_char:

            type\_pairs.append(text\_pairs[i] + text\_pairs[i + 1])

            i = i + 2

        else:

            type\_pairs.append(text\_pairs[i] + text\_pairs[i + 1])

            i = i + 2

    print("Text Pairs:")

    print(type\_pairs, "\n")

def get\_row\_index(char: str):

    if char == "I" or char == "J":

        char = "I/J"

    if char in row\_1:

        return {

            "index": row\_1.index(char),

            "row\_num": 1

        }

    if char in row\_2:

        return {

            "index": row\_2.index(char),

            "row\_num": 2

        }

    if char in row\_3:

        return {

            "index": row\_3.index(char),

            "row\_num": 3

        }

    if char in row\_4:

        return {

            "index": row\_4.index(char),

            "row\_num": 4

        }

    if char in row\_5:

        return {

            "index": row\_5.index(char),

            "row\_num": 5

        }

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

    if len(plain\_text) == 0: return

    if len(key) == 0: return

    plain\_text = plain\_text.upper()

    plain\_text = "".join(plain\_text.split())

    key = key.upper()

    key = "".join(key.split())

    for i 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))}%")

    playfair\_matrix(key)

    get\_text\_pairs(plain\_text, encrypted\_pairs)

    encrypted\_text = ""

    for i in encrypted\_pairs:

        first\_info = get\_row\_index(i[0])

        second\_info = get\_row\_index(i[1])

        first\_row, first\_col = first\_info["row\_num"]-1, first\_info["index"]

        second\_row, second\_col = second\_info["row\_num"]-1, second\_info["index"]

        if first\_row == second\_row:

            encrypted\_text += (matrix[first\_row][(first\_col + 1) % 5])

            encrypted\_text += (matrix[second\_row][(second\_col + 1) % 5])

        elif first\_col == second\_col:

            encrypted\_text += (matrix[(first\_row + 1) % 5][first\_col])

            encrypted\_text += (matrix[(second\_row + 1) % 5][second\_col])

        else:

            encrypted\_text += (matrix[first\_row][second\_col])

            encrypted\_text += (matrix[second\_row][first\_col])

    return encrypted\_text

def playfair\_cipher\_decrypt(cipher\_text: str, key: str):

    if len(cipher\_text) == 0: return

    if len(key) == 0: return

    cipher\_text = cipher\_text.upper()

    cipher\_text = "".join(cipher\_text.split())

    key = key.upper()

    key = "".join(key.split())

    get\_text\_pairs(cipher\_text, decrypted\_pairs)

    decrypted\_text = ""

    for i in decrypted\_pairs:

        first\_info = get\_row\_index(i[0])

        second\_info = get\_row\_index(i[1])

        first\_row, first\_col = first\_info["row\_num"]-1, first\_info["index"]

        second\_row, second\_col = second\_info["row\_num"]-1, second\_info["index"]

        if first\_row == second\_row:

            decrypted\_text += (matrix[first\_row][(first\_col - 1) % 5])

            decrypted\_text += (matrix[second\_row][(second\_col - 1) % 5])

        elif first\_col == second\_col:

            decrypted\_text += (matrix[(first\_row - 1) % 5][first\_col])

            decrypted\_text += (matrix[(second\_row - 1) % 5][second\_col])

        else:

            decrypted\_text += (matrix[first\_row][second\_col])

            decrypted\_text += (matrix[second\_row][first\_col])

    return decrypted\_text

original\_message = "The key is hidden under the door"

original\_key = "domestic"

print("\nOriginal Message: ", original\_message)

encrypted\_message = playfair\_cipher\_encrypt(original\_message, original\_key)

print("Encrypted Message: ", encrypted\_message)

print()

decrypted\_message = playfair\_cipher\_decrypt(encrypted\_message, original\_key)

print("Decrypted Message: ", decrypted\_message)

OUTPUT:



