import numpy as

Today 11:17 PM No category

```
import numpy as np


def generate_playfair_matrix(keyword):
    """Generates the 5x5 Playfair cipher
matrix based on the keyword."""
    keyword = keyword.lower().replace("j",
"i")  # Playfair typically combines I/J
    matrix = []
    seen = set()

    for char in keyword +
"abcdefghiklmnopqrstuvwxyz":
        if char not in seen:
            seen.add(char)
```

```
            matrix.append(char)

    return np.array(matrix).reshape(5, 5)

def find_position(matrix, letter):
    """Finds the row and column of a letter
in the Playfair matrix."""
    row, col = np.where(matrix == letter)
    return row[0], col[0]

def process_pairs(text):
    """Splits text into pairs for Playfair
cipher, adding 'X' if needed."""
    text = text.lower().replace("j", "i")
    pairs = []
    i = 0
```

```
    while i < len(text):
        a = text[i]
        b = text[i + 1] if i + 1 < len(text) else
'x'

        if a == b:  # Same letter pair, insert 'X'
            pairs.append(a + 'x')
            i += 1
        else:
            pairs.append(a + b)
            i += 2

    return pairs

def encrypt_pair(pair, matrix):
    """Encrypts a pair of letters using
Playfair cipher rules."""
```

```
    r1, c1 = find_position(matrix, pair[0])
    r2, c2 = find_position(matrix, pair[1])

    if r1 == r2:  # Same row: Shift right
        return matrix[r1, (c1 + 1) % 5] +
matrix[r2, (c2 + 1) % 5]
    elif c1 == c2:  # Same column: Shift
down
        return matrix[(r1 + 1) % 5, c1] +
matrix[(r2 + 1) % 5, c2]
    else:  # Rectangle rule
        return matrix[r1, c2] + matrix[r2, c1]

def decrypt_pair(pair, matrix):
    """Decrypts a pair of letters using
Playfair cipher rules."""
    r1, c1 = find_position(matrix, pair[0])
```

```
    r2, c2 = find_position(matrix, pair[1])

    if r1 == r2:  # Same row: Shift left
        return matrix[r1, (c1 - 1) % 5] +
matrix[r2, (c2 - 1) % 5]
    elif c1 == c2:  # Same column: Shift up
        return matrix[(r1 - 1) % 5, c1] +
matrix[(r2 - 1) % 5, c2]
    else:  # Rectangle rule
        return matrix[r1, c2] + matrix[r2, c1]

def encrypt(text, matrix):
    """Encrypts a message using the Playfair
cipher."""
    pairs = process_pairs(text)
    return "".join(encrypt_pair(pair, matrix)
for pair in pairs)
```

```
def decrypt(text, matrix):
    """Decrypts a message using the Playfair
cipher."""
    pairs = process_pairs(text)
    return "".join(decrypt_pair(pair, matrix)
for pair in pairs)

# Example Usage
keyword = input("Enter keyword: ")
matrix =
generate_playfair_matrix(keyword)
print("\nPlayfair Matrix:")
print(matrix)

while True:
    choice = input("\nDo you want to
```

```
(E)ncrypt or (D)ecrypt? ").strip().lower()

    if choice == 'e':
        plaintext = input("Enter plaintext:
").replace(" ", "").lower()
        encrypted_text = encrypt(plaintext,
matrix)
        print(f"Encrypted: {encrypted_text}")
    elif choice == 'd':
        ciphertext = input("Enter ciphertext:
").replace(" ", "").lower()
        decrypted_text = decrypt(ciphertext,
matrix)
        print(f"Decrypted: {decrypted_text}")
    else:
        print("Invalid choice. Exiting...")
        break
```