Lab: 3 **Date:2082/10/13**

Title: Write a Program to encrypt and decrypt the user input message and key using Play Fair cipher

The Playfair Cipher is a digraph substitution cipher used in classical cryptography. Instead of encrypting single letters like the Caesar Cipher, it encrypts pairs of letters (digraphs), making it more secure against simple frequency analysis.

Purpose:

The Playfair Cipher is used to encrypt messages so that they are harder to decode than simple substitution ciphers. Decryption reverses the process, transforming the encrypted message (ciphertext) back into the original message (plaintext) using the same key square.

How it works:

1. **Key Square Preparation**:
   - Fill a 5×5 grid with the letters of the keyword (without repetition).
   - Fill the remaining cells with the other letters of the alphabet (combining 'I' and 'J')
2. **Encryption Rules**:
   - If a pair has the same letter (e.g., LL), insert a filler like X between them.
   - Apply the following rules for each pair:
     1. **Same row**: Replace each letter with the letter immediately to its right
     2. **Same column**: Replace each letter with the letter immediately below
     3. **Different row and column**: Replace each letter with the letter in the same row but column of the other letter
3. **Decryption Rules**:
   - Reverse the encryption rules:

Key Features:

- Encrypt letter pairs (digraphs), no single letters.
- Repeated letters in a pair use a filler (like 'X').

Advantages:

- More secure than simple substitution ciphers since it encrypts letter pairs.
- It is harder to break using basic frequency analysis.

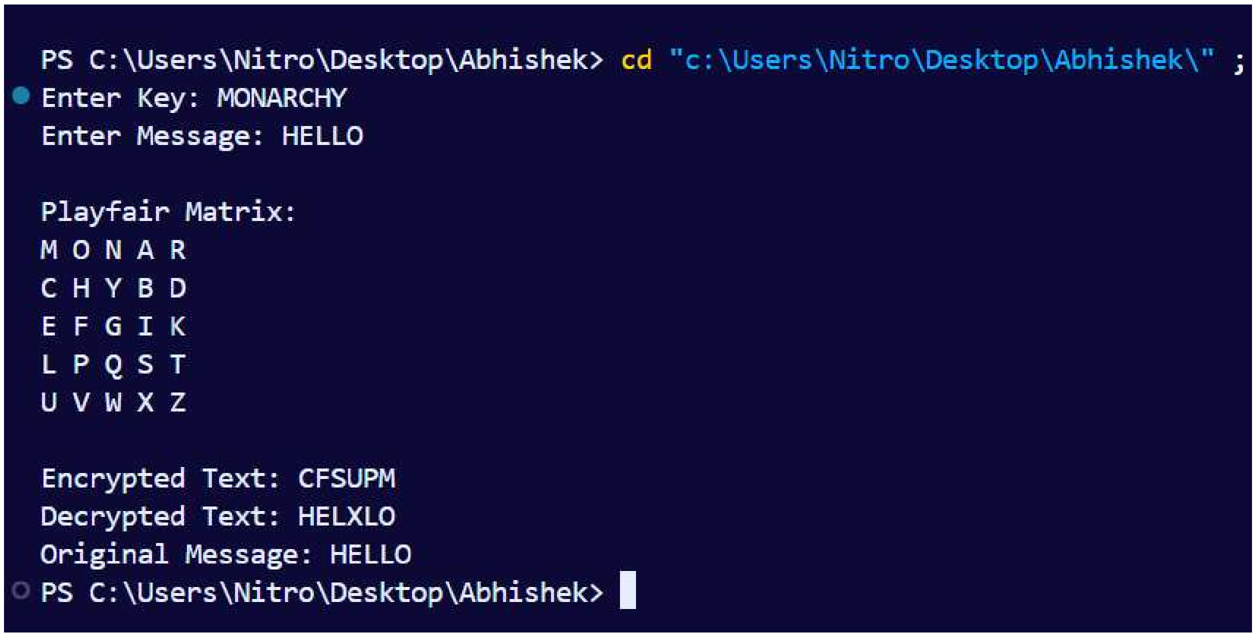
Source code:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

char matrix[5][5];

// Create 5x5 Playfair matrix from key
void createMatrix(char key[]) {
    int used[26] = {0};
    int i, k = 0;

    for (i = 0; key[i]; i++) {
        key[i] = toupper(key[i]);
        if (key[i] == 'J') key[i] = 'I';
        if (!used[key[i] - 'A']) {
            matrix[k / 5][k % 5] = key[i];
            used[key[i] - 'A'] = 1;
            k++;
        }
    }

    for (i = 0; i < 26; i++) {
        if (i + 'A' == 'J') continue;
        if (!used[i]) {
            matrix[k / 5][k % 5] = i + 'A';
            k++;
        }
    }
}

// Print the Playfair matrix
void printMatrix() {
    int i, j;
    printf("\nPlayfair Matrix:\n");
    for (i = 0; i < 5; i++) {
        for (j = 0; j < 5; j++) {
            printf("%c ", matrix[i][j]);
        }
        printf("\n");
    }
}

// Find row and column of a character
void findPos(char c, int *row, int *col) {
    int i, j;
    if (c == 'J') c = 'I';
    for (i = 0; i < 5; i++)
        for (j = 0; j < 5; j++)
```

```
            if (matrix[i][j] == c) {
                *row = i;
                *col = j;
                return;
            }
}

// Encrypt the text and store result in 'encrypted'
void encryptText(char text[], char encrypted[]) {
    int i, r1, c1, r2, c2, k = 0;
    int len = strlen(text);
    for (i = 0; i < len; i += 2) {
        findPos(text[i], &r1, &c1);
        findPos(text[i + 1], &r2, &c2);

        if (r1 == r2) {
            encrypted[k++] = matrix[r1][(c1 + 1) % 5];
            encrypted[k++] = matrix[r2][(c2 + 1) % 5];
        } else if (c1 == c2) {
            encrypted[k++] = matrix[(r1 + 1) % 5][c1];
            encrypted[k++] = matrix[(r2 + 1) % 5][c2];
        } else {
            encrypted[k++] = matrix[r1][c2];
            encrypted[k++] = matrix[r2][c1];
        }
    }
    encrypted[k] = '\0';
}

// Decrypt the text and store result in 'decrypted'
void decryptText(char text[], char decrypted[]) {
    int i, r1, c1, r2, c2, k = 0;
    int len = strlen(text);
    for (i = 0; i < len; i += 2) {
        findPos(text[i], &r1, &c1);
        findPos(text[i + 1], &r2, &c2);

        if (r1 == r2) {
            decrypted[k++] = matrix[r1][(c1 + 4) % 5];
            decrypted[k++] = matrix[r2][(c2 + 4) % 5];
        } else if (c1 == c2) {
            decrypted[k++] = matrix[(r1 + 4) % 5][c1];
            decrypted[k++] = matrix[(r2 + 4) % 5][c2];
        } else {
            decrypted[k++] = matrix[r1][c2];
            decrypted[k++] = matrix[r2][c1];
        }
    }
    decrypted[k] = '\0';
}
```

```c
// Remove extra X used as padding between repeated letters
void removeX(char text[]) {
    char result[100];
    int i, j = 0;
    for (i = 0; text[i]; i++) {
        if (i > 0 && text[i] == 'X' && text[i-1] == text[i+1])
            continue;
        result[j++] = text[i];
    }
    result[j] = '\0';
    strcpy(text, result);
}
int main() {
    char key[50], msg[50], text[100], encrypted[100], decrypted[100];
    int i, j = 0;

    printf("Enter Key: ");
    scanf("%s", key);

    printf("Enter Message: ");
    scanf("%s", msg);

    // Convert message to uppercase
    for (i = 0; msg[i]; i++)
        msg[i] = toupper(msg[i]);

    // Prepare message for digraphs (insert X if letters repeat)
    j = 0;
    for (i = 0; msg[i]; i++) {
        text[j++] = msg[i];
        if (msg[i] == msg[i + 1])
            text[j++] = 'X';
    }
    if (j % 2 != 0) text[j++] = 'X';
    text[j] = '\0';

    // Create Playfair matrix
    createMatrix(key);

    // Print the matrix
    printMatrix();

    // Encrypt
    encryptText(text, encrypted);
    printf("\nEncrypted Text: %s\n", encrypted);

    // Decrypt
    decryptText(encrypted, decrypted);
    printf("Decrypted Text: %s\n", decrypted);

    // Remove padding X to get original
    removeX(decrypted);
    printf("Original Message: %s\n", decrypted);

    return 0;
}
```

Output:



```
PS C:\Users\Nitro\Desktop\Abhishek> cd "c:\Users\Nitro\Desktop\Abhishek\" ;
Enter Key: MONARCHY
Enter Message: HELLO

Playfair Matrix:
M O N A R
C H Y B D
E F G I K
L P Q S T
U V W X Z

Encrypted Text: CFSUPM
Decrypted Text: HELXLO
Original Message: HELLO
PS C:\Users\Nitro\Desktop\Abhishek>
```

Conclusion

The Playfair Cipher demonstrates a more advanced form of substitution-based encryption by operating on pairs of letters rather than individual characters. Through the use of a keyword-based key square and positional rules, it introduces structured encryption and decryption processes. Although it is not secure by modern cryptographic standards, the Playfair Cipher effectively illustrates key-based symmetric encryption, digraph substitution, and the importance of algorithmic rules in strengthening message confidentiality.