Reg no: 210701092

Experiment no: 1b
Date: 30/1/24

Playfair Cipher

AIM:
To Write the C program to perform playfair Cipher for Both Encryption And Decryption

ALGORITHM:
Step 1: START
Step 2: Get the Key And Text from the User
Step 3: Create the Matrix Function Using the Key
Step 4: Perform Encryption Using the Matrix key
Step 5: ~~Perform Decryption from~~ In Decryption function Retrive the Message from the Encryption And Decrypt the Message
Step 6: Print the Encryption And Decryption Message
Step 7: STOP

Exp 2 : Playfair Cipher

Code:

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

void change_to_lowercase(char plain[], int ps) {
for (int i = 0; i < ps; i++) {
        if (plain[i] > 64 && plain[i] < 91)
            plain[i] += 32;
    }
}

int remove_all_spaces(char* plain, int ps) {
int i, count = 0;
    for (i = 0; i < ps; i++)
        if (plain[i] != ' ')
            plain[count++] = plain[i];
    plain[count] = '\0';
    return count;
}

void generate_key(char key[], int ks, char keyT[5][5]) {
int i, j, k, flag = 0, *dicty;

    dicty = (int*)calloc(26, sizeof(int));

    for (i = 0; i < ks; i++) {
        if (key[i] != 'j')
            dicty[key[i] - 97] = 2;
}
    dicty['j' - 97] = 1;

    i = 0;
    j = 0;
    for (k = 0; k < ks; k++) {
        if (dicty[key[k] - 97] == 2) {
dicty[key[k] - 97] -= 1;
            keyT[i][j] = key[k];
            j++;
            if (j == 5) {
                i++;
                j = 0;
            }
        }
    }
    for (k = 0; k < 26; k++) {
        if (dicty[k] == 0) {
            keyT[i][j] = (char)(k + 97);
j++;
            if (j == 5) {
                i++;
                j = 0;
            }
        }
    }
}
```

```
void searching(char keyT[5][5], char a, char b, int arr[]) {
int i, j;

    if (a == 'j')
        a = 'i';
    else if (b == 'j')
b = 'i';

    for (i = 0; i < 5; i++) {
        for (j = 0; j < 5; j++) {
            if (keyT[i][j] == a) {
                arr[0] = i;
                arr[1] = j;
            }
            else if (keyT[i][j] == b) {
arr[2] = i;
                arr[3] = j;
            }
        }
    }
}

int mod5(int a) {
if (a < 0)
        a += 5;
    return (a % 5);
}

void encrypt(char str[], char keyT[5][5], int ps) {
int i, a[4];
    for (i = 0; i < ps; i += 2) {
        searching(keyT, str[i], str[i + 1], a);
        if (a[0] == a[2]) {
            str[i] = keyT[a[0]][(a[1] + 1) % 5];
            str[i + 1] = keyT[a[0]][(a[3] + 1) % 5];
}
        else if (a[1] == a[3]) {
            str[i] = keyT[(a[0] + 1) % 5][a[1]];
            str[i + 1] = keyT[(a[2] + 1) % 5][a[1]];
}
        else {
            str[i] = keyT[a[0]][a[3]];
            str[i + 1] = keyT[a[2]][a[1]];
        }
    }
}

void decrypt(char str[], char keyT[5][5], int ps) {
int i, a[4];
    for (i = 0; i < ps; i += 2) {
        searching(keyT, str[i], str[i + 1], a);
        if (a[0] == a[2]) {
            str[i] = keyT[a[0]][mod5(a[1] - 1)];
            str[i + 1] = keyT[a[0]][mod5(a[3] - 1)];
}
        else if (a[1] == a[3]) {
            str[i] = keyT[mod5(a[0] - 1)][a[1]];
            str[i + 1] = keyT[mod5(a[2] - 1)][a[1]];
}
        else {
            str[i] = keyT[a[0]][a[3]];
            str[i + 1] = keyT[a[2]][a[1]];
}
```

```
    }
}

void playfair_cipher(char str[], char key[], int mode) {
char ps, ks, keyT[5][5];

    ks = strlen(key);
    ks = remove_all_spaces(key, ks);
change_to_lowercase(key, ks);

    ps = strlen(str);
    change_to_lowercase(str, ps);
    ps = remove_all_spaces(str, ps);

    generate_key(key, ks, keyT);

    if (mode == 0) {
        encrypt(str, keyT, ps);
        printf("Cipher Text: %s\n", str);
    }
    else {
        decrypt(str, keyT, ps);
        printf("Deciphered Text: %s\n", str);
}
}

int main() {
    char str[200], key[200];
int mode;

    printf("Enter key: ");
scanf("%[^\n]s", &key);
printf("Enter Text: ");
scanf("\n");
    scanf("%[^\n]s", &str);

    printf("Enter Mode (0 for Encryption, 1 for Decryption): ");
scanf("%d", &mode);

    playfair_cipher(str, key, mode);

    return 0;
}
```

Output 1:

```
Enter key: wheatz
Enter Text: wars
Enter Mode (0 for Encryption, 1 for Decryption): 0
Cipher Text: htny
```

Output 2:

```
Enter key: wheatz
Enter Text: htny
Enter Mode (0 for Encryption, 1 for Decryption): 1
Deciphered Text: wars
```