

EXP NO:2

DATE:

PLAYFAIR CIPHER

Aim: To implement an encryption algorithm using Playfair Cipher technique.

Algorithm:

- Step 1: "Algorithm" (as the key) and "ulroaliocvrX" (as the encrypted text).
- Step 2: Remove spaces and convert to lowercase.
- Step 3: Create a 5x5 key table based on the modified key.
- Step 4: Apply Playfair Cipher decryption to the encrypted text using the generated key table.
- Step 5: Display the deciphered text.

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h> #define
SIZE 30
void toLowerCase(char plain[], int ps)
{
    int i;
    for (i = 0; i < ps; i++) {
        if (plain[i] > 64 && plain[i] < 91)
            plain[i] += 32;
    }
}
int removeSpaces(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 generateKeyTable(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 search(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 decrypt(char str[], char keyT[5][5], int ps) {
    int i, a[4];    for (i = 0; i < ps; i += 2) {
search(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 decryptByPlayfairCipher(char str[], char key[])
{    char ps, ks, keyT[5][5];
ks = strlen(key);    ks =
removeSpaces(key, ks);
toLowerCase(key, ks);    ps =
strlen(str);    toLowerCase(str,
ps);    ps = removeSpaces(str,
ps);

```

```

    generateKeyTable(key, ks, keyT);

```

```

    decrypt(str, keyT, ps);
}

```

```

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

    strcpy(key, "Thrisha");
    printf("Key text: %s\n", key);
    strcpy(str, "ulroaliocvrX");
    printf("Plain text: %s\n", str);

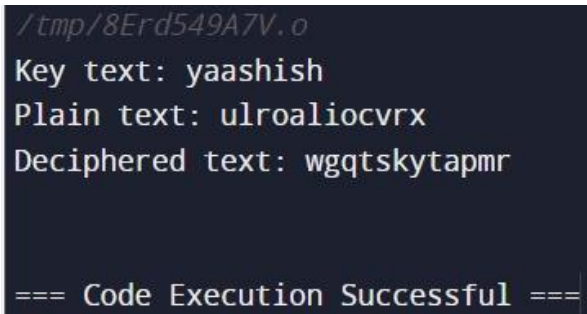
    decryptByPlayfairCipher(str, key);

```

```
printf("Deciphered text: %s\n", str);

return 0;
}
```

Output:

A terminal window with a dark background and light-colored text. The output shows the execution of a program that takes a key and a plain text as input and produces a deciphered text. The text is as follows:

```
/tmp/8Erd549A7V.o
Key text: yaashish
Plain text: ulroaliocvrX
Deciphered text: wgqtskytapmr

=== Code Execution Successful ===
```

Result: