```
#include <stdio.h>

#include <string.h>

#include <ctype.h>


#define SIZE 5


Void generateKeyMatrix(char key[], char keyMatrix[SIZE][SIZE]) {

  Int alphabet[26] = {0};

  Int x = 0, y = 0;


  For (int I = 0; key[i] != '\0'; i++) {

    Char ch = tolower(key[i]);

    If (ch == 'j') ch = 'I'; // Treat 'I' and 'j' as the same

    If (isalpha(ch) && !alphabet[ch – 'a']) {

      keyMatrix[x][y++] = ch;

      alphabet[ch – 'a'] = 1;

      if (y == SIZE) {

        y = 0;

        x++;

      }

    }

  }


  For (char ch = 'a'; ch <= 'z'; ch++) {

    If (ch == 'j') continue;

    If (!alphabet[ch – 'a']) {
      keyMatrix[x][y++] = ch;

      if (y == SIZE) {

        y = 0;

        x++;

      }

    }

  }

}


Void formatText(char text[]) {

  Int len = strlen(text), index = 0;

  Char formatted[100] = {0};

  For (int I = 0; I < len; i++) {

    If (isalpha(text[i])) {

      Formatted[index++] = tolower(text[i] == ‘j’ ? ‘I’ : text[i]);

    }

  }


  For (int I = 0; I < index; I += 2) {

    If (I + 1 < index && formatted[i] == formatted[I + 1]) {

      Memmove(&formatted[I + 1], &formatted[i], index – i);

      Formatted[I + 1] = ‘x’;

      Index++;

    }

  }
    If (index % 2 != 0) {
        Formatted[index++] = ‘x’;
    }

    Formatted[index] = ‘\0’;
    Strcpy(text, formatted);
}

Void findPosition(char keyMatrix[SIZE][SIZE], char ch, int *row, int *col) {
    For (int I = 0; I < SIZE; i++) {
        For (int j = 0; j < SIZE; j++) {
            If (keyMatrix[i][j] == ch) {
                *row = I;
                *col = j;
                Return;
            }
        }
    }
}

Void playfairCipher(char text[], char keyMatrix[SIZE][SIZE], int encrypt) {
    For (int I = 0; text[i] != ‘\0’; I += 2) {
        Int r1, c1, r2, c2;
        findPosition(keyMatrix, text[i], &r1, &c1);
        findPosition(keyMatrix, text[I + 1], &r2, &c2);
        if (r1 == r2) {
          text[i] = keyMatrix[r1][(c1 + encrypt + SIZE) % SIZE];
          text[I + 1] = keyMatrix[r2][(c2 + encrypt + SIZE) % SIZE];
        } else if (c1 == c2) {
          Text[i] = keyMatrix[(r1 + encrypt + SIZE) % SIZE][c1];
          Text[I + 1] = keyMatrix[(r2 + encrypt + SIZE) % SIZE][c2];
        } else {
          Text[i] = keyMatrix[r1][c2];
          Text[I + 1] = keyMatrix[r2][c1];
        }
      }
    }

Int main() {
  Char key[100], text[100], keyMatrix[SIZE][SIZE];


  Printf(“Enter key: “);
  Fgets(key, sizeof(key), stdin);
  Key[strcspn(key, “\n”)] = ‘\0’;


  Printf(“Enter text: “);
  Fgets(text, sizeof(text), stdin);
  Text[strcspn(text, “\n”)] = ‘\0’;


  generateKeyMatrix(key, keyMatrix);
  formatText(text);
```

```
    printf(“Choose an option:\n1. Encrypt\n2. Decrypt\n”);
    int choice;
    scanf(“%d”, &choice);

    if (choice == 1) {
        playfairCipher(text, keyMatrix, 1);
        printf(“Encrypted text: %s\n”, text);
    } else if (choice == 2) {
        playfairCipher(text, keyMatrix, -1);
        printf(“Decrypted text: %s\n”, text);
    } else {
        Printf(“Invalid choice\n”);
    }

    Return 0;
}
```