```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#define MOD 26
int charToInt(char c) {
    return toupper(c) - 'A';
}
char intToChar(int i) {
    return (i % MOD + MOD) % MOD + 'A';
}
int modInverse(int det) {
    det = (det % MOD + MOD) % MOD;
    for (int i = 1; i < MOD; i++) {
        if ((det * i) % MOD == 1)
            return i;
    }
    return -1;  // No inverse
}
void multiplyMatrix(int key[2][2], int vector[2], int result[2]) {
    for (int i = 0; i < 2; i++) {
        result[i] = 0;
        for (int j = 0; j < 2; j++) {
            result[i] += key[i][j] * vector[j];
        }
        result[i] %= MOD;
    }
}
int invertMatrix(int key[2][2], int inverse[2][2]) {
    int det = key[0][0]*key[1][1] - key[0][1]*key[1][0];
    int invDet = modInverse(det);
    if (invDet == -1) return 0;
    inverse[0][0] =  key[1][1] * invDet % MOD;
    inverse[0][1] = -key[0][1] * invDet % MOD;
    inverse[1][0] = -key[1][0] * invDet % MOD;
    inverse[1][1] =  key[0][0] * invDet % MOD;
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            inverse[i][j] = (inverse[i][j] + MOD) % MOD;

    return 1;
```

```
}
void hillEncrypt(char *plaintext, int key[2][2], char *ciphertext) {
    int len = strlen(plaintext);
    if (len % 2 != 0) {
        plaintext[len] = 'X';  // padding with 'X'
        plaintext[len + 1] = '\0';
        len++;
    }
    for (int i = 0; i < len; i += 2) {
        int vector[2] = { charToInt(plaintext[i]), charToInt(plaintext[i+1]) };
        int result[2];
        multiplyMatrix(key, vector, result);
        ciphertext[i] = intToChar(result[0]);
        ciphertext[i+1] = intToChar(result[1]);
    }
    ciphertext[len] = '\0';
}
void hillDecrypt(char *ciphertext, int key[2][2], char *plaintext) {
    int inverse[2][2];
    if (!invertMatrix(key, inverse)) {
        printf("Key matrix not invertible. Decryption not possible.\n");
        return;
    }
    int len = strlen(ciphertext);
    for (int i = 0; i < len; i += 2) {
        int vector[2] = { charToInt(ciphertext[i]), charToInt(ciphertext[i+1]) };
        int result[2];
        multiplyMatrix(inverse, vector, result);
        plaintext[i] = intToChar(result[0]);
        plaintext[i+1] = intToChar(result[1]);
    }
    plaintext[len] = '\0';
}
int main() {
    char plaintext[100], ciphertext[100], decrypted[100];
    int key[2][2] = {
        {3, 3},
        {2, 5}
    };
```

```c
int main() {
    char plaintext[100], ciphertext[100], decrypted[100];
    int key[2][2] = {
        {3, 3},
        {2, 5}
    };
    printf("Enter plaintext (only A-Z): ");
    scanf("%s", plaintext);
    hillEncrypt(plaintext, key, ciphertext);
    printf("Encrypted: %s\n", ciphertext);
    hillDecrypt(ciphertext, key, decrypted);
    printf("Decrypted: %s\n", decrypted);
    return 0;
}
```

input

```
Enter plaintext (only A-Z): ramana
Encrypted: ZIKYNA
Decrypted: RAMANA


...Program finished with exit code 0
Press ENTER to exit console.
```