#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <ctype.h>

#define MAX 100

// Function to sort key and store order

void getKeyOrder(char\* key, int\* order) {

int len = strlen(key);

char tempKey[MAX];

strcpy(tempKey, key);

for (int i = 0; i < len; i++) {

order[i] = i;

}

// Bubble sort based on key characters

for (int i = 0; i < len-1; i++) {

for (int j = 0; j < len-i-1; j++) {

if (tempKey[j] > tempKey[j+1]) {

// Swap in tempKey

char t = tempKey[j];

tempKey[j] = tempKey[j+1];

tempKey[j+1] = t;

// Swap in order

int tmp = order[j];

order[j] = order[j+1];

order[j+1] = tmp;

}

}

}

}

// Encryption

void encrypt(char\* plaintext, char\* key, char\* ciphertext) {

int keyLen = strlen(key);

int textLen = strlen(plaintext);

int row = (textLen + keyLen - 1) / keyLen; // ceiling

char matrix[row][keyLen];

memset(matrix, 'X', sizeof(matrix)); // Fill with padding character

// Fill matrix row-wise

int k = 0;

for (int i = 0; i < row && k < textLen; i++) {

for (int j = 0; j < keyLen && k < textLen; j++) {

matrix[i][j] = plaintext[k++];

}

}

// Get order of columns

int order[keyLen];

getKeyOrder(key, order);

// Read matrix column-wise using order

k = 0;

for (int i = 0; i < keyLen; i++) {

int col = order[i];

for (int j = 0; j < row; j++) {

ciphertext[k++] = matrix[j][col];

}

}

ciphertext[k] = '\0';

}

// Decryption

void decrypt(char\* ciphertext, char\* key, char\* plaintext) {

int keyLen = strlen(key);

int textLen = strlen(ciphertext);

int row = (textLen + keyLen - 1) / keyLen;

char matrix[row][keyLen];

// Get order of columns

int order[keyLen];

getKeyOrder(key, order);

// Fill matrix column-wise

int k = 0;

for (int i = 0; i < keyLen; i++) {

int col = order[i];

for (int j = 0; j < row; j++) {

matrix[j][col] = ciphertext[k++];

}

}

// Read row-wise to get plaintext

k = 0;

for (int i = 0; i < row; i++) {

for (int j = 0; j < keyLen; j++) {

plaintext[k++] = matrix[i][j];

}

}

plaintext[k] = '\0';

}

int main() {

char plaintext[MAX], key[MAX], ciphertext[MAX], decrypted[MAX];

printf("Enter the key (no spaces): ");

scanf("%s", key);

printf("Enter the plaintext (no spaces): ");

scanf("%s", plaintext);

encrypt(plaintext, key, ciphertext);

printf("Encrypted text: %s\n", ciphertext);

decrypt(ciphertext, key, decrypted);

printf("Decrypted text: %s\n", decrypted);

return 0;

}

