#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define MOD 26

int modInverse(int a, int m) {

a = a % m;

for (int x = 1; x < m; x++) {

if ((a \* x) % m == 1)

return x;

}

return -1;

}

void multiplyMatrix(int key[2][2], int vec[2], int result[2]) {

for (int i = 0; i < 2; i++) {

result[i] = (key[i][0] \* vec[0] + key[i][1] \* vec[1]) % MOD;

}

}

int inverseMatrix(int key[2][2], int invKey[2][2]) {

int det = key[0][0]\*key[1][1] - key[0][1]\*key[1][0];

det = (det % MOD + MOD) % MOD;

int invDet = modInverse(det, MOD);

if (invDet == -1) return 0;

invKey[0][0] = ( key[1][1] \* invDet) % MOD;

invKey[0][1] = (-key[0][1] \* invDet + MOD) % MOD;

invKey[1][0] = (-key[1][0] \* invDet + MOD) % MOD;

invKey[1][1] = ( key[0][0] \* invDet) % MOD;

return 1;

}

void encrypt(char\* plaintext, int key[2][2], char\* ciphertext) {

int vec[2], result[2], i = 0;

while (plaintext[i] != '\0') {

vec[0] = toupper(plaintext[i]) - 'A';

vec[1] = (plaintext[i+1] != '\0') ? toupper(plaintext[i+1]) - 'A' : 'X' - 'A';

multiplyMatrix(key, vec, result);

ciphertext[i] = result[0] + 'A';

ciphertext[i+1] = result[1] + 'A';

i += 2;

}

ciphertext[i] = '\0';

}

void decrypt(char\* ciphertext, int key[2][2], char\* plaintext) {

int invKey[2][2];

if (!inverseMatrix(key, invKey)) {

printf("Key matrix is not invertible.\n");

return;

}

int vec[2], result[2], i = 0;

while (ciphertext[i] != '\0') {

vec[0] = toupper(ciphertext[i]) - 'A';

vec[1] = toupper(ciphertext[i+1]) - 'A';

multiplyMatrix(invKey, vec, result);

plaintext[i] = result[0] + 'A';

plaintext[i+1] = result[1] + 'A';

i += 2;

}

plaintext[i] = '\0';

}

int main() {

char plaintext[100], ciphertext[100], decrypted[100];

int key[2][2] = {{3, 3}, {2, 5}};

printf("Enter plaintext (A-Z only): ");

fgets(plaintext, sizeof(plaintext), stdin);

plaintext[strcspn(plaintext, "\n")] = '\0';

if (strlen(plaintext) % 2 != 0) {

strcat(plaintext, "X");

}

encrypt(plaintext, key, ciphertext);

decrypt(ciphertext, key, decrypted);

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

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

return 0;

}

