#include <iostream>

#include <fstream>

#include <stdint.h>

using namespace std;

unsigned int key[4] = { 0xFDA5,0xD54E,0xFC00,0xB55A };

#define BLOCK\_SIZE 8

void xtea\_encipher(unsigned int num\_rounds, uint32\_t v[2], uint32\_t const key[4]) {

unsigned int i;

uint32\_t v0 = v[0], v1 = v[1], sum = 0, delta = 0x9E3779B9;

for (i = 0; i < num\_rounds; i++) {

v0 += (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]);

sum += delta;

v1 += (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum >> 11) & 3]);

}

v[0] = v0; v[1] = v1;

}

void xtea\_decipher(unsigned int num\_rounds, uint32\_t v[2], uint32\_t const key[4]) {

unsigned int i;

uint32\_t v0 = v[0], v1 = v[1], delta = 0x9E3779B9, sum = delta \* num\_rounds;

for (i = 0; i < num\_rounds; i++) {

v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum >> 11) & 3]);

sum -= delta;

v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]);

}

v[0] = v0; v[1] = v1;

}

void StringCrypt(char\* inout, int len, bool encrypt)

{

for (int i = 0; i < len / BLOCK\_SIZE; i++)

{

if (encrypt)

xtea\_encipher(32, (uint32\_t\*)(inout + (i \* BLOCK\_SIZE)), key);

else

xtea\_decipher(32, (uint32\_t\*)(inout + (i \* BLOCK\_SIZE)), key);

}

if (len % BLOCK\_SIZE != 0)

{

int mod = len % BLOCK\_SIZE;

int offset = (len / BLOCK\_SIZE) \* BLOCK\_SIZE;

char data[BLOCK\_SIZE];

memcpy(data, inout + offset, mod);

if (encrypt)

xtea\_encipher(32, (uint32\_t\*)data, key);

else

xtea\_decipher(32, (uint32\_t\*)data, key);

memcpy(inout + offset, data, mod);

}

}

int main()

{

char vvod[16];

cout << "Vvedite text: ";

gets\_s(vvod);

int len = sizeof(vvod);

StringCrypt(vvod, len, true);

cout << "Encrypted string: ";

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

cout << vvod[i];

cout << endl;

StringCrypt(vvod, len, false);

cout << "Decrypted string: ";

cout << vvod << endl;

}

