Source Code

// Name: Preston Knibbe

//

// Purpose: To demostrate string functions by using basic crytopgraphy methods

//

#include <iostream>

#include <fstream>

#include <string>

#include <cstdlib>

using namespace std;

char substitution\_cipher(string cipher\_key, char char\_to\_encrypt);

char reverse\_substitution\_cipher(string cipher\_key, char char\_to\_encrypt);

string EncryptString(string &cipher\_key, string string\_to\_be\_encrypted);

string DecryptString(string &cipher\_key, string string\_to\_be\_decrypted);

void RotateCipherKey(string &cipher\_key);

void DisplayFile(string filename, string cipher\_key);

void EncryptFile(string cipher\_key, string filename\_from, string filename\_to);

void DecryptFile(string cipher\_key, string filename\_from, string filename\_to);

int main()

{

string cipher\_key = "qwertyuiopasdfghjklzxcvbnm";

string cipher\_key2 = "qwertyuiopasdfghjklzxcvbnm";

EncryptFile(cipher\_key, "test.txt", "test-encrypted.txt");

DecryptFile(cipher\_key2, "test-encrypted.txt", "test-ed.txt");

DisplayFile("test.txt", cipher\_key2);

DisplayFile("test-encrypted.txt", cipher\_key2);

DisplayFile("test-ed.txt", cipher\_key2);

return 0;

}

// Rotate the cipher key. Example: abcdef becames bcdefa

//

// Note: Call by reference

void RotateCipherKey(string &cipher\_key)

{

char first\_char = cipher\_key[0];

for (int x = 1; x <= 25; x++) {

cipher\_key[x-1] = cipher\_key[x];

}

cipher\_key[25] = first\_char;

}

// Perform a substitution cipher on a single character

// using the specified cipher key

// See reference document for implementation details.

char SubstitutionCipher(string cipher\_key, char char\_to\_encrypt)

{

char finder[26] = {'a', 'b', 'c', 'd', 'e', 'f', 'g',

'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',

's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};

char encrypted\_char;

for(int x = 0; x <= 26; x++) {

if (finder[x] == char\_to\_encrypt) {

encrypted\_char = cipher\_key[x];

}

}

return encrypted\_char;

}

// Perform a "reverse" substitution cipher on a single character

// using the specified cipher key

// See reference document for implementation details.

char ReverseSubstitutionCipher(string cipher\_key, char char\_to\_decrypt)

{

char finder[26] = {'a', 'b', 'c', 'd', 'e', 'f', 'g',

'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',

's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};

char decrypted\_char;

for (int x = 0; x <= 26; x++) {

if (cipher\_key[x] == char\_to\_decrypt) {

decrypted\_char = finder[x];

}

}

return decrypted\_char;

}

// Encrypt String and return it

// You will use the SubstitutionCipher() function to encrypt the

// individual characters

//

// Note: You have to call RotateCipherKey() after each time you encrypt

// a character.

string EncryptString(string &cipher\_key, string string\_to\_be\_encrypted)

{

string encrypted\_string = "";

for (int x = 0; x < string\_to\_be\_encrypted.length(); x++) {

encrypted\_string += SubstitutionCipher(cipher\_key, string\_to\_be\_encrypted[x]);

RotateCipherKey(cipher\_key);

}

return encrypted\_string;

}

// Decrypt String and return it

// You will use the ReverseSubstitutionCipher() function to decrypt the

// individual characters

//

// Note: You have to call RotateCipherKey() after each time you encrypt

// a character.

string DecryptString(string &cipher\_key, string string\_to\_be\_decrypted)

{

string decrypted\_string = "";

for (int x = 0; x < string\_to\_be\_decrypted.length(); x++) {

decrypted\_string += ReverseSubstitutionCipher(cipher\_key, string\_to\_be\_decrypted[x]);

RotateCipherKey(cipher\_key);

}

return decrypted\_string;

}

// Display file specified by the filname parameter

void DisplayFile(string filename, string cipher\_key)

{

string input;

ifstream infile;

infile.open (filename.c\_str());

if(!infile)

{

cout << "Can not open input file " + filename << endl;

exit(0);

}

while( getline(infile, input))

{

cout << EncryptString(cipher\_key, input) << endl;

}

infile.close();

}

// Encrypt the specified file using the specified cipher key and

// write the output to a different file

// This function is complete

void EncryptFile(string cipher\_key, string filename\_from, string filename\_to)

{

string input;

ifstream infile;

ofstream outfile;

infile.open (filename\_from.c\_str());

outfile.open (filename\_to.c\_str());

if(!infile)

{

cout << "Can not open input file " + filename\_from << endl;

exit(0);

}

if(!outfile)

{

cout << "Can not open Output file " + filename\_to << endl;

exit(0);

}

while( getline(infile, input))

{

outfile << input << endl;

}

infile.close();

outfile.close();

}

// Decrypt the specified file using the specified cipher key and

// write the output to a different file

// This function is complete

void DecryptFile(string cipher\_key, string filename\_from, string filename\_to)

{

string input;

ifstream infile;

ofstream outfile;

infile.open (filename\_from.c\_str());

outfile.open (filename\_to.c\_str());

if(!infile)

{

cout << "Can not open input file " + filename\_from << endl;

exit(0);

}

if(!outfile)

{

cout << "Can not open Output file " + filename\_to << endl;

exit(0);

}

while( getline(infile, input))

{

outfile << DecryptString(cipher\_key, input) << endl;

}

infile.close();

outfile.close();

}

Output

zoac fn qbvh tgezyt qrbnejozp

zjaiqf fzkjm krt utaek ujx xbcbfw jqjugus xom wrelqlb

fcbfhybnd huf waj lipx qzbkjvj

zoac fn qbvh tgezyt qrbnejozp

zjaiqf fzkjm krt utaek ujx xbcbfw jqjugus xom wrelqlb

fcbfhybnd huf waj lipx qzbkjvj

this is some random gibberish

please check all lines for proper encrypt and decrypt

hopefully you got this working

Process returned 0 (0x0) execution time : 0.020 s

Press any key to continue.