COMP-165-0837

Merlin Newcomb

Streams Assignment

Problem #1

Code:

/\*

\* File encyption is the science of writing the contents of a file in a secret code.

\* Your encryption program should work like a filter, reading the contents of one file, modifying the data into a code, then writing the coded contents out to a second file.

\* The second file will be a version of the first file, but written in a secret code.

\* Although there are complex encyption techniques, you should come up with a simple one of your own.

\* For example, you could read the first file one character at a time, and add 10 to the ASCII code of each character before it is written to the second file.

\*/

#include <iostream>

#include <string>

#include <fstream>

#include <sstream>

#include <cstdio>

using namespace std;

// Performs XOR encryption/decryption on specified file name with key.

void xorFile(const string& fileName, const string& key);

// Encrypts file using xorFile, then renames to encrypted.txt.

void encrypt(const string& fileName, const string& key);

// Decrypts file using xorFile, then renames to decrypted.txt.

void decrypt(const string& fileName, const string& key);

// Was also thinking of base64 encoding encrypted files to make them readable, but I'd sadly also have to implement that myself

int main(void)

{

string fileName;

string key;

cout << "Enter file name: " << flush;

getline(cin, fileName);

cout << "Enter key: " << flush;

getline(cin, key);

fstream file(fileName);

if (!file.is\_open())

{

cerr << "Error: File cannot be opened." << endl;

return -1;

}

cout << "encrypt: encrypt file with given key." << endl;

cout << "decrypt: decrypt file with given key." << endl;

cout << "exit: exit the program." << endl;

while (true)

{

string userInput;

getline(cin, userInput);

if (userInput == "encrypt")

{

encrypt(fileName, key);

break;

}

else if (userInput == "decrypt")

{

decrypt(fileName, key);

break;

}

else if (userInput == "exit")

{

cout << "Exiting." << endl;

break;

}

cout << "Invalid input." << endl;

}

}

void xorFile(const string& fileName, const string& key)

{

ifstream inFile(fileName, fstream::binary);

string inFileContent(istreambuf\_iterator<char>(inFile), {}); // String range constructed from istreambuf\_iterator, iterates through raw characters until end of stream

ofstream outFile("xorfile.txt");

string workingKey(key);

while (workingKey.size() < inFileContent.size())

{

workingKey += key;

}

for (size\_t i = 0; i < inFileContent.size(); i++)

{

inFileContent.at(i) ^= workingKey.at(i);

}

outFile.write(inFileContent.c\_str(), inFileContent.size());

}

void encrypt(const string& fileName, const string& key)

{

xorFile(fileName, key);

if (rename("xorfile.txt", "encrypted.txt"))

{

cout << "Renaming xorfile.txt failed. Does file exist?" << endl;

}

if (remove(fileName.c\_str()))

{

cout << "Error deleting " << fileName << endl;

}

}

void decrypt(const string& fileName, const string& key)

{

xorFile(fileName, key);

if (rename("xorfile.txt", "decrypted.txt"))

{

cout << "Renaming xorfile.txt failed. Does file exist?" << endl;

}

if (remove(fileName.c\_str()))

{

cout << "Error deleting " << fileName << endl;

}

}

Sample message.txt:

This is a test message!

Secret stuff here

asldk;l;cmleoiasdcfkjl

the quick brown fox jumps over the lazy dog

0000000000000000000000000000000000000000000

yes that's supposed to be a bunch of 0

Output:

