Lab 3: Caesar Cipher

Date assigned: Feb 4th, 2016 Due Date: Feb 11th, 2016

Problem Description:

To send secret messages we use ciphers. A cipher is an algorithm for performing encryption and decryption. Encryption transforms a plain text into an illegible code, which can be transformed back to the original message using systematic steps of decryption. In this lab we will be implementing Caesar's Cipher.

Caesar cipher is a shift cipher. To encrypt a message each letter in the message is replaced by another letter some fixed number of positions down the alphabet. For example when we shift alphabet by 2, and *wraparound* then each alphabet in top row will be encoded as the corresponding alphabet in the bottom row:

A	B	C	D	E	F

Y	Z	A	B	C	D

This may look mundane but the encoded messages are hard to read, for example:

WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

But if we move each letter back 3 positions we get:

THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

The sender encrypts the message and instead of sending the "shift", sends the most frequent alphabet character in the original message. The receiver computes the most common encrypted alphabet character in the cyphered text and takes the difference from the most common alphabet character in the original message to figure out the shift. The sender can retrieve the original message from the cypher by reversing the shift.

Note: In the case of multiple letters tying as the most frequent in the message, the alphabet character corresponds to the first occurrence of one of the most frequent characters.

What You Need to Do:

You are provided with two C files: "main.c" and "lab3.c". Your primary job is to ***complete unfinished functions in "lab3.c"*** such that we can both encrypt messages and decrypt them.

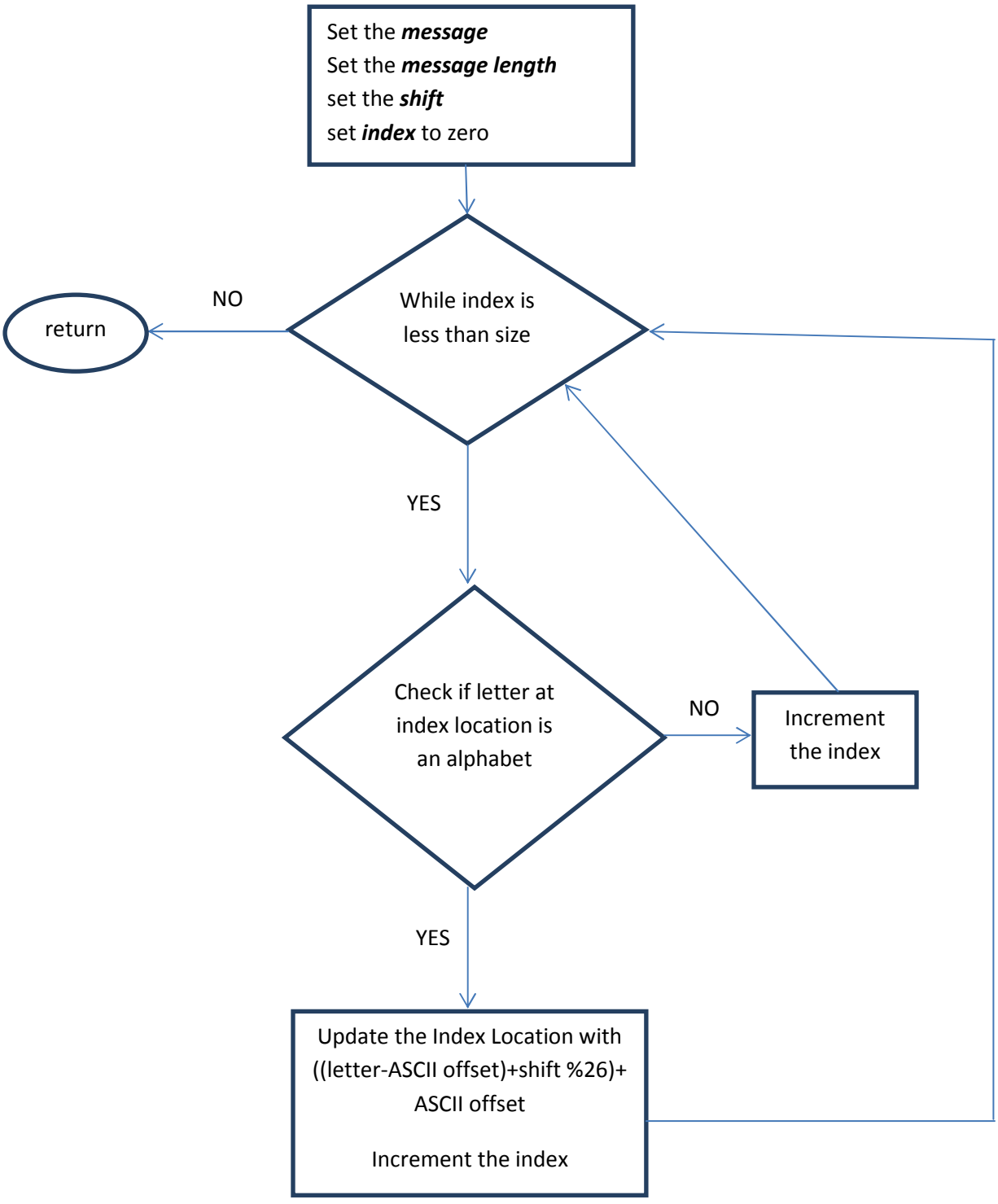
Grading:

This lab is worth 15 points. We will test your program for three main functionalities of the lab: encryption, finding the most frequent alphabet and decryption, each of these functionalities carries 5 point

Deliverables:

For this assignment, you need to complete the unimplemented functions in the lab3.c file provided to you.
You will commit to your assembla repo both the lab3.c file and the main.c file.
Place these files in a new folder called “assignment3” and commit. Your “assignment3” folder should contain only these two files.
Do not change the lab3.c and main.c file to .cpp files.

The Encryption Step Flow Diagram:



Computing the most frequent alphabet:

We want to transmit the message produced from encryption step, but if we give out the shift as it is, this will compromise our encryption scheme. Therefore, we transmit the *most frequent alphabet* of the original message separately to the recipient.

We will build a histogram of alphabets that will record the frequency of the each alphabet in the message. For the data below

A	A	B	A	A	C	C	A	A	C

Histogram is

Index	0	1	2
Frequency	6	1	3

Where indexes map to alphabet

Index	Value
0	A
1	B
2	C

The lengths of both the data arrays and histogram array are known in advance.

Furthermore, the data values will be made to map to the indexes of histogram array. This done by representing A as 0 , B as 1 … and Z as 25. The flow chart below elaborates the histogram construction.

The largest value in the histogram array should be sent along with the message to help the decoder.

Decryption of the Encoded Message:

Here we will reuse the code we wrote for the encryption. First we will compute the most common alphabet in the encrypted message, using the histogram approach. Secondly we subtract most common alphabet of original message from encrypted alphabet to compute the shift. Finally we decrypt the message by re-encrypting it using the reverse-shift.

Build Histogram: