Sam Dinkelman

10/5/18

CIS 247

Lab 1 Report

Introduction:

For this Lab we were tasked with writing two functions, the first finds a substring within a string and the second replaces part of the string with the substring. For this code I used the gcc options -Wall and -o to view all warnings and specify the name of the output file.

Process:

In order to accomplish this task I started working on the Find function with a for loop:

for(i = 0; i < strlen(str); i++){

This for loop iterates through the data.txt file as long as the integer i is less than the length of the string variable str which is this case is the data.txt file.

Next I wrote an if statement to test if the string (str) at the index integer (i) was equal to the substring (substr) at the first element of the array [0] which would be the first character of the substring(substr).

if(str[i] == substr[0]){

flag == 0;

Then if this statement is accurate then it will set a variable called flag equal to zero which we will see again in a later if statement.

After this I used another for statement to evaluate the substring(substr), when the if statement above is true it will use this for loop to iterate through the substring(substr) as long as the integer(j) is less than the length of the substring(substr). I started the index integer(j) at one since this for loop would only be entered if the first character of the substring(substr) had already been found in the if statement above.

for(j = 1; j < strlen(substr); j++){

Next I setup another if statement that compares the string(str) at the index integer(i) plus the index integer(j) to the substring(substr) to see if they are not equal to each other. My reasoning for having integer(i) plus integer(j) is that the index integer(i) will already be a the position of the start of the substring(substr) if we have gotten this far, and the index integer(j) will increment as long as the string length(strlen) is greater than the integer index(j). Without this critical index addition the code would simply find the first char of the substring(substr) and not increment through the entirety of the substring(substr). If this statement is satisfied then the variable flag will be set to one and negative one will be returned.

if(str[i+j] != substr[j]){

flag = 1;

The next if statement checks if the flag variable is equal to zero and returns the index integer(i). This will return the position of the index integer(i) if the condition of the earlier if statement that checks to see if the string(str) is equal to the substring(substr) at the given index integer(i) position is evaluated to accurate.

if(flag == 0){

return i;

The last statement in my Find function is a return statement that just returns -1, this will happen if the first if statement doesn’t evaluate to accurate when it checks to see if the string(str) is equal to the substring(substr) at the first character of the substring(substr).

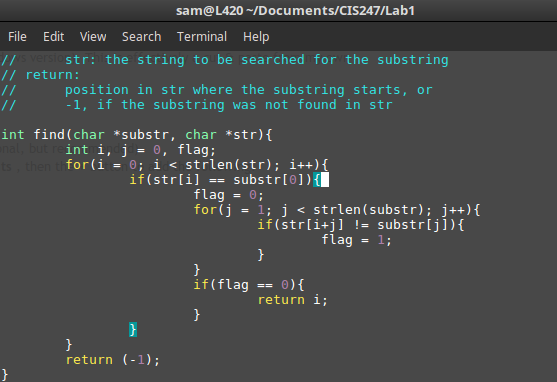
In my Replace function I start off with a for loop that initializes an index integer(i) and iterates as long as the index integer is less than the string length(strlen) of the substring(substr). This ensures that it goes through every character that is stored in that location in memory that the substring(substr) is pointing to.

for(int i = 0; i < strlen(substr); i++){

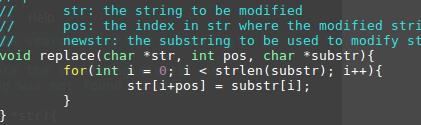
The next line sets the value of the string(str) at position of index integer(i) plus the position integer(pos) equal to the substring(substr) position at index integer(i).

str[i+pos] = substr[i];

The rest of the code is all within the function main which was already written for us.

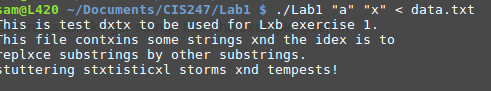
Here is my Find function:

Here is my Replace function:

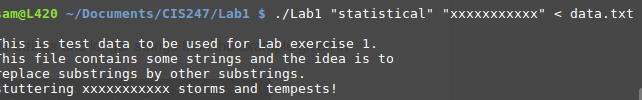


Testing:

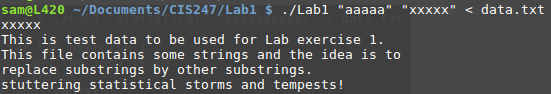
So for testing I wanted to try shorter strings and longer strings than the example given in the lab so I tried replacing a with x



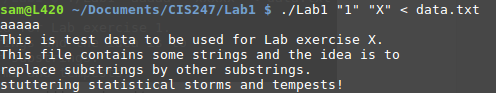
I also tried longer strings like replacing statistical with xxxxxxxxxxx



Words with the same letters in a row to check if the comparisons were working properly and stepping through all the letters in the given substring. There weren’t any words with multiple of the same letter in a row so I added an aaaaa string to the data.txt file at the beginning.



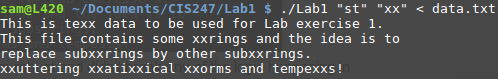
Lastly I wanted to check if numbers would be replaced too.



All of my tests proved my program robust and didn’t break any of the logic happening in my program.

Results

The following screenshot shows that my program completes the requirements laid out in the Lab 1 description, replacing “st” with “xx”



Conclusion

Based on my results and the intent of the project I found this to be a good exercise in demonstrating some of the core fundamentals of the C language. It made for a decent challenge to start to program using C but wasn’t overwhelmingly difficult. I’m very proud of my little program that seems to be robust and handle the test cases I gave to it very well.

References and Acknowledgement

Over the course of this assignment I received guidance from many of my peers as to how to handle the functions we were asked to write. However all of this code is my own work and I worked hard on my own before collaborating with peers to work together to find the correct approach.