Programming Assignment #5

This is another short assignment to give you more practice with NUL-terminated strings, functions, iteration, and other fun stuff. The program you write will scan and manipulate strings. There are four functions that you need to implement to complete the assignment.

Function	Description
`int count_tabs(const char *string);`	Given a string, count the number of tabs in the string and return the count. This is about 5 lines of code.

`int substitute_char(char *string,`
`char old_char,`
`char new_char);`
	Given a string, substitute `old_char` with `new_char`. Returns the number of substitutions. This is about 6 lines of code.

`void calculate_lengths(const char *string,`
`int tabsize,`
`int *string_length,`
`int *display_length);`
	Given a string and a tabsize, calculate the length of the string and the print length of the string. The print length is the number of characters that will be required to display the string after the tabs are expanded into *tabsize* spaces. This is about 5 lines of code.

`int count_words(const char *string);`	Given a string, count the number of words and return the count. Words in a string are delimited by *whitespace* characters which are any of the following: space, newline, tab. The sample driver shows numerous examples. This is about 12 lines of code.

Notes:

- You can assume that all of the strings are properly NUL-terminated so the length isn't passed to the functions.
- There is a `mystrlen` function implemented in **`scantext.c`**, which is included for you to use, if you need.
- The function, `calculate_lengths`, takes two integer pointers. These pointers are used to return the values to the calling function (`main`, in this case).
- You should work on the functions in the order listed, since they are listed roughly by the order of difficulty.
- If you wish to write additional "helper" functions to use, they must be marked with the **`static`** keyword (like the `mystrlen` function that is provided.)
- You must not include any header files in your `scantext.c` file. **Including any header files will result in an automatic 0.**

Details

You are given a file called **`driver.c`**, which includes the main function with several test cases for you to use. There are four functions prototyped in the driver, one for each of the functions you are to write. As always, you must implement these functions *exactly as prototyped.* A partial file is provided for you to start with. A sample command line to compile this assignment looks like this:

```
gcc -O -Wall -Wextra -Werror -ansi -pedantic driver.c scantext.c -o scantext
```

This will compile and link both files and produce an executable file named **`scantext.exe`** on Windows. This assignment will not require you to include any header files in your code, so don't. All of the code that you write must be in `scantext.c`, since you will not be turning in any other files.

Some or all of the functions will be doing some sort of looping. You can use either a **for** loop, **while** loop, or **do** loop and you can use subscripting or pointers.

What to submit
You must submit **`scantext.c`** to the appropriate submission server.

Refer to the web page for this assignment for any additional details on how to submit this assignment. **Do not submit any other files than the ones listed.**

Files	Description
scantext.c	The C file. This file contains all of the source code for the program, properly formatted and commented as discussed in class.

If you've forgotten how to submit files, the details about how to submit are posted on the course website and in the syllabus. Failure to follow the instructions will result in a poor score on the assignment.

Make sure your name and other info is on all documents (paper and electronic).